Visual Servo Control on a Humanoid Robot

by
Michael Bosongo Bombile

A dissertation submitted to the Department of Electrical Engineering,
University of Cape Town, in fulfillment of the requirements
for the degree of Master of Science in Engineering.

Supervisor: Prof. Martin Braae

Cape Town, June 2015

Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the degree of Master of Science in Engineering in the University of Cape Town. It has not been submitted before for any degree or examination in any other university.

Signature of Author .....

Cape Town
30 June 2015

Abstract

This thesis deals with the control of a humanoid robot based on visual servoing. It seeks to confer a degree of autonomy to the robot in the achievement of tasks such as reaching a desired position, tracking or/and grasping an object. The autonomy of humanoid robots is considered as crucial for the success of the numerous services that this kind of robots can render with their ability to associate dexterity and mobility in structured, unstructured or even hazardous environments.

To achieve this objective, a humanoid robot is fully modeled and the control of its locomotion, conditioned by postural balance and gait stability, is studied. The presented approach is formulated to account for all the joints of the biped robot. As a way to conform the reference commands from visual servoing to the discrete locomotion mode of the robot, this study exploits a reactive omnidirectional walking pattern generator and a visual task Jacobian redefined with respect to a floating base on the humanoid robot, instead of the stance foot. The redundancy problem stemming from the high number of degrees of freedom coupled with the omnidirectional mobility of the robot is handled within the task priority framework, allowing thus to achieve configuration dependent sub-objectives such as improving the reachability, the manipulability and avoiding joint limits.

Beyond a kinematic formulation of visual servoing, this thesis explores a dynamic visual approach and proposes two new visual servoing laws. Lyapunov theory is used first to prove the stability and convergence of the visual closed loop, then to derive a robust adaptive controller for the combined robot-vision dynamics, yielding thus an ultimate uniform bounded solution. Finally, all proposed schemes are validated in simulation and experimentally on the humanoid robot NAO.

To the Master of time and circumstances
To my beloved family
To my best friend Lyd

Acknowledgments

Although words will never suffice, I would like to express my sincere gratitude to many people who in some way or another contributed to the completion of this dissertation.

I would like to thank first my supervisor Prof. Martin Braae, for his outstanding guidance along this research journey and for giving me the freedom to explore my own ideas.

I would like to thank also my colleagues and office-mates for the motivation they had instilled in me throughout the course of this work.

I am deeply thankful to Prof. Meissner and his family for making me feeling at home in Cape Town.

My special gratitude goes to Lydie Kabulo, for the love, patience, and support she gave me along these years of studies, and also for reminding me that life is not only about Robotics.

Finally, I will be eternally indebted to my parents, my brothers and sisters for their love, sacrifice, support, and encouragement. *Du fond de mon coeur, je vous dis merci.*

Michael Bombile

Contents

List of Figures

List of Tables

Notations

Abbreviations

CoM	—	Center of Mass
CoP	—	Center of Pressure
D-H	—	Denavit-Hartenberg
DoF	—	Degree of Freedom
DSP	—	Double Support Phase
EE	—	End-Effector
IBVS	—	Image based Visual Servoing
LIPM	—	Linear Inverted Pendulum Model
LMPC	—	Linear Model Predictive Control
PBVS	—	Position based Visual Servoing
ROWPG	—	Reactive Omnidirectional Walking Pattern Generator
SSP	—	Single Support Phase
ViSP	—	Visual Servoing Platform
ZMP	—	Zero Moment Point

Symbols

${}^{c}\mathbf{R}_{w}$	—	Rotation matrix from world frame to camera frame
${}^{c}\mathbf{t}_{w}$	—	translation vector from world frame to camera frame
$\mathcal{F}_{w}$	—	World reference frame
${}^{c}\mathbf{M}_{w}$	—	Homogeneous transformation from world frame to camera frame
$\boldsymbol{e}(t)$	—	Task function
$\boldsymbol{\xi}$	—	Current feature vector
$\boldsymbol{\xi}^{*}$	—	Desired feature vector
$\frac{\partial \boldsymbol{e}}{\partial t}$	—	Variation of task function due to object motion

$\mathbf{L}_{\xi}$	—	Interaction matrix related to the feature $\boldsymbol{\xi}$
${}^{c}V$	—	Camera velocity expressed in camera frame
${}^{c}W_{e}$	—	Twist transformation matrix from end-effector's to camera frame
${}^{e}\mathbf{J}_{q}$	—	Jacobian matrix of the robot expressed in end-effector frame
$\widehat{\mathbf{L}_{e}}$	—	Approximate or estimate of the interaction matrix related to $\boldsymbol{e}(t)$
$\widehat{\mathbf{J}_{e}^{\dagger}}$	—	Pseudo-inverse of the approximate visual task Jacobian matrix
$\mathbf{L}_{\theta\mathbf{u}}$	—	Interaction matrix related to the feature $\theta\mathbf{u}$
$[\mathbf{u}]_{\times}$	—	Skew symmetric matrix related to vector $\mathbf{u}$
$\mathbf{q}$	—	vector of the robot's joints
$\boldsymbol{X}_{i}$	—	Vector of coordinates of operational space i
$\mathbf{M}(\mathbf{q})$	—	Inertia matrix of the robot
$\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$	—	Matrix of Coriolis and Centrifugal effects
$\mathbf{G}(\mathbf{q})$	—	Vector of Gravity force on the robot
$\boldsymbol{\Gamma}$	—	Vector of applied torques on the robot
$\mathbf{t}_{b}$	—	Translation vector of the humanoid robot's base
$\boldsymbol{\phi}_{b}^{T}$	—	Vector of orientation parameters of the humanoid robot's base
$\mathcal{T}_{st}$	—	Reaction wrench acting on the humanoid robot's stance foot
$\mathcal{A}_{\mathcal{F}_{st}}^{T}(\mathbf{q})$	—	Matrix of constraints acting on the humanoid robot's stance foot
$\mathcal{T}_{imp_{st}}$	—	Impulse torque acting on the humanoid foot during impact phase
$\hat{c}_{k}^{h}$	—	State space vector of the Linear Inverted Pendulum Model (LIPM)
$\widehat{C}_{k+1}^{h}$	—	State space vector of the LIPM over the prediction horizon
P_{k+1}^{h}	—	Actual ZMP position over the prediction horizon
$P_{k+1}^{h\_ref}$	—	Reference ZMP position over the prediction horizon
${}^{w}\boldsymbol{F}_{k+1}^{h}$	—	Reference footsteps positions over the prediction horizon
${}^{w}\boldsymbol{\Theta}_{k+1}^{ref}$	—	Reference footsteps orientations over the prediction horizon
$\mathcal{L}(\boldsymbol{\epsilon},t)$	—	Candidate Lyapunov function
λ	—	Eigen value
σ	—	Singular value
${}^{c}\widehat{V}_{off}$	—	Visual feed-forward control input

Chapter 1

Introduction

1.1 Background

In recent years, robotics research has shown a growing interest for mobile robots, able to overcome a well-controlled environments where industrial robots operate, to venture into unstructured and dynamic environments. The motivation behind the interest in this kind of robots is mainly due to numerous and various applications offered by the latter. For example, assistance to elderly or sick people, applications in hazardous environments for humans, surveillance, rescue operations after disaster, military applications, underwater and space exploration, etc.
Among these mobile robots, biped humanoid robots have the characteristic to associate mobility with manipulation and can easily blend into environments designed for humans. However, success in accomplishing their mission depends heavily on their level of autonomy, which is still far from expectations.

In order to increase their autonomy, in addition to proprioceptive sensors which provide information on their internal state, these robots are currently equipped with different types of exteroceptive sensors (cameras, ultrasound sensor, etc.) to allow them to perceive their environment, interact with it and adapt to it when necessary. In this perspective, the contribution of vision, in reference to that of human, becomes very significant. However, if it is true that perception is necessary to increase robots autonomy, it is not sufficient in itself. It is still required that the information gathered by the sensors be intelligently used to achieve the desired behavior. In other words, the information needs to be processed, decisions made, and actions planned and executed.

Thus, the use of visual perception to control robots has been widely investigated. From reported results, two complementary approaches emerge. The first, rather deliberative, is to use visual information to plan trajectories that the robot must follow in order to accomplish the desired task. In this approach however, the accuracy is closely linked to that of the sensor and to the available models of the scene and the robot itself. This approach is best suited for quasi-

static environments. In case of a changing environment, the time-consuming or computationally demanding planning phase has to be repeated in real-time. The second approach is for its part reactive; the perception is directly related to the action without the need for a prior planning phase. It is in this approach that "*visual servoing*" falls, which consists in using visual information to close the loop of a robotic system in order to control it. This technique is more accurate and provides more robustness to modeling and calibration errors [2][3].

In the literature, one can find several studies involving vision in tasks achievement on humanoid robots. However, few of them have focused on the visual servoing, which is the subject of this thesis. From the reported studies on visual servoing of humanoid robots, we can distinguish three types of applications. The first type focuses on the upper body part of the robot, with emphasis on manipulation tasks [4, 5, 6, 7, 8, 9, 10, 11], visual tracking tasks (gazing) [12, 13, 11, 14]. This problem is generally solved as visual servoing of a classical robot manipulator [3, 15].

The second category is mainly concerned with locomotion tasks, while assuming fixed upper body limbs. Unlike visual servoing of the robot's upper-body part, whose motion is continuous, visual servoing of the robot's lower-body part is more challenging due essentially to the bipedal nature of the locomotion. This form of locomotion is discrete and can only be realized under particular constraints satisfying the postural balance and the gait stability [16]. In that respect, common locomotion module often relies on a gait planner, which requires knowledge beforehand of the goal position and orientation in order to plan footsteps. With these considerations, the visual measurements, in these approaches, are generally used to reconstruct the three dimensional information such as the relative poses (positions and orientations) between the robot and the target. Then, the recovered information is sent to the walking module, which plans and executes the footsteps and the robot trajectories according to a stability indicator, usually the Zero-Moment-Point (ZMP) [17, 18].

These kinds of approaches are convenient for positioning tasks. However, for tracking tasks or any other task not expressible in terms of positioning, their performances decrease, because all commands issued by the visual controller have to be converted in positioning commands for the gait planner. Moreover, though mitigated by the feedback, the effect of sensitivity to modeling and calibration errors due to the 3D reconstruction is also a concern.

It is worth mentioning however the promising approach, purely reactive, proposed in [19, 20] and [21, 22]. It allows overriding the classical footsteps planner, by using the automatic footsteps placement pattern generator developed by *Herdt et al.*, [23]. Under the assumption that the robot's center of mass is rigidly linked to the camera, the two dimensional image measurements were used to generate velocity commands to be followed by the humanoid's center of mass.

Similarly, a velocity based control was recently proposed in [24] for a navigation task, where the locomotion module was modeled as a mobile unicycle robot.

The third kind of application combines visual servoing of both the upper body and lower body parts of the robot, which results in a "whole body task". For instance, a grasping task while walking was presented in [25]. In this kind of application, a coordination between the motions of both parts is necessary. Moreover, all possible motion of the upper part has to be accounted for in the balance and stabilization scheme.

1.2 Objectives and Approach

In this thesis, the objective is to control humanoid robots based on visual servoing in order to achieve some autonomous behaviors. In particular, positioning tasks, tracking tasks as well as grasping tasks are considered. This technique will be first discussed in terms of the current literature before being applied to the humanoid robot NAO.

To perform this kind of reactive control on a humanoid robot, many challenges must be considered. Indeed, the robot has hybrid dynamics due to its bipedal locomotion. This implies that continuity in task space will be associated with continuous and discrete modes in the joints space, while the usual visual servoing formulation assumes continuity in the joints space. From a control perspective this means that the visual system has to generate continuous control commands for a robot whose dynamics switches at each step. In addition, the generated control commands must ensure balance and stability for a robot which can only exert, through a reduced support surface, unilateral forces on the ground. Otherwise, the humanoid would tip over. Also, task redundancy due to the high number of degrees of freedom, combined with omnidirectional mobility is an issue that must be addressed.

The fact that the common kinematic approach to visual servoing results in slow response, often with poor dynamic performances [26], one additional objective in this thesis is to consider a dynamic approach to visual servoing in attempt to enhance the overall system performances under this "reactive" control method.

In order to achieve these objectives, visual servoing methods are first surveyed, with a particular attention to its applications on humanoid robots. The dynamic model of a humanoid robot, showing its hybrid and constrained nature, is derived and its locomotion problem is studied.
From this theoretical framework, as an approach to visual servoing of humanoid robots in general and then of the humanoid NAO in particular, we will use essentially the two dimensional visual

servoing technique for its greater robustness to modeling and calibration errors. When needed, the depth of the scene will be estimated. We will formulate the kinematics of the robot so as to ensure continuity in task space despite discrete modes at the joints level and to simplify the inclusion of balance and stability constraints.

To ensure a reactive locomotion compatible with reference commands from the vision system, we will adopt *Herdt*'s [23, 27] online gait generator that will have to be extended to overcome the predetermination of rotation. Unlike the approaches [19, 20, 21, 28, 22] and [24], which have assumed the camera to be rigidly linked to the robot's center of mass, we will seek a general formulation where the neck joints as well as those of other upper body limbs are free to move and will be, therefore, included in the solution. The associated redundancy problem will be addressed within the task priority framework [29], where secondary tasks will be projected onto the null-space of the main task in order to optimize the robot configuration.

Regarding dynamic visual servoing, our approach will consist of improving first the dynamic of reference commands generated by the vision system, then in designing a controller in order to enhance the robot tracking performances for the generated commands.

The practical viability of the proposed theoretical solutions, will be explored first in simulation to provide initial results. Then, the solutions will be experimentally validated on an actual humanoid robot, the humanoid robot NAO, which will have to perform the considered tasks autonomously.

1.3 Scope and Thesis Outlines

Performing visual servoing-based tasks involves many disciplines such as computer vision, kinematics and dynamics, control theory, etc. For example, in a positioning task with respect to an object, it is necessary, in all acquired images, to distinguish and extract information related to the target object from its background. Once this information is collected, based on the kinematics or even the dynamics of the robot, a control law must then be designed to determine the robot motions required to accomplish the given task.

In this thesis, however, we will assume the image processing already done and we will limit ourselves to the use of kinematics and dynamics of the robot for the design of control laws.
Besides the introduction, this dissertation is divided into two main parts, corresponding respectively to a kinematic and dynamic approach to the visual servoing problem.

Chapter 2 presents the fundamental concepts related to visual servoing and provides a state of the art in control of humanoid robots based on visual servoing. We are particularly interested

in the choice of visual information, in the control structures and the types of control laws used.

Chapter 3 presents modeling aspects of a humanoid robot in general and discusses the locomotion control problem with a focus on predictive control based pattern generators, particularly the online pattern generator proposed by *Herdt*. Since it is limited to translational motions, the reactivity of *Herdt*'s pattern generator is extended to rotational velocities.

Chapter 4 presents the solution proposed in this work to the problem of visual servoing of a humanoid robot. We first reformulate the kinematics of the robot to account for stability constraints and solve the redundancy problem. Then, using 2D visual information, we apply this formulation to positioning, tracking and grasping tasks.

Chapter 5 describes first the humanoid robot chosen as the experimental platform, namely the humanoid robot NAO V4.0. The forward, the inverse and the velocity kinematic models are explicitly derived in order to provide all variables necessary to implement visual servoing. Then, extensive simulations are carried out to explore the viability of the proposed approach. Finally, experiments are conducted on an actual humanoid robot NAO to show the effectiveness of our design to perform the assigned tasks.

Chapter 6 proposes our approach to dynamic visual servoing, particularly for indirect visual servoing. The reference velocity is computed to yield a second order decrease of the task function with a PD-type controller which could give faster and damped responses in the task space. Then a robust adaptive controller is design to compensate for the system's dynamics and to ensure the tracking of the generated references.

In Chapter 7, as a way to validate our proposed dynamic visual servoing approach, simulations are carried out to compare its performances with those of the classical visual servoing approach. Then, the proposed scheme is experimentally applied to the dynamic control of the humanoid NAO's head to achieve visual positioning and visual tracking of a moving target.

Chapter 8 presents the general conclusions and gives some recommendations for future studies.

Part I

Visual Servoing on Humanoid Robot: the Kinematic Approach

Chapter 2

General Concepts on Visual Servoing

This chapter provides a general overview on fundamental concepts of visual servoing[1] and reviews some of its reported applications on humanoid robots. The first section recalls the mechanism of image formation. Then, some definitions related to visual servoing are presented and itself is modeled within the general framework of task function [35]. We focus on the two main techniques namely: position based and image based visual servoing. And we also give a brief overview of alternative techniques. Finally, prior works applying visual servoing to control humanoid robots are reviewed. More specifically, their objectives, the type of visual information and the type of command used are discussed.

2.1 Principle of Visual Servoing

In visual servoing, the principle consists of specifying the robotic task to be performed in terms of image measurements, which are extracted in real-time from acquired images of the target through image processing and computer vision algorithms. Hence, based on these image measurements, the robot's loop is closed and a controller is designed to minimize an appropriate error function by moving the robot/camera [36], [30], [3]. This is illustrated in Figure 2.1 showing how visual servoing improves the robustness to modeling errors and enhances flexibility and autonomy of a robot, which was blindly performing tasks [37], [31].

2.2 Classification

Depending on the configuration between the camera and the robot, the control scheme, or most importantly the definition of the task (error) function, one can classify visual servoing techniques in different categories [38], [30].

[1]For more details on this discipline, excellent survey and tutorials covering the basics and advanced concepts can be found in [30], [31], [32], [33], [3, 15] and [34].

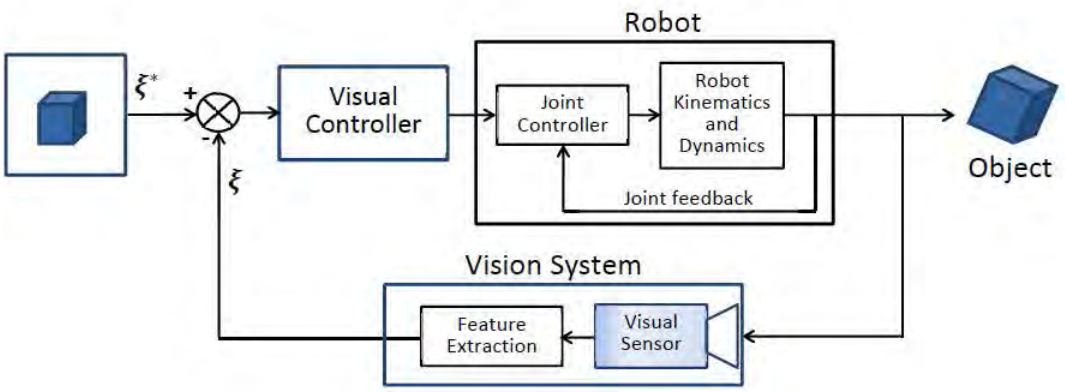


Figure 2.1: Block diagram summarizing visual servoing principle. It shows how information about the robot end-effector is fed back to a visual controller which computes control commands that minimize the error between the current image parametrized by ξ and the desired image parametrized by ξ^d.

2.2.1 Camera – Robot Configuration

In visual servoing, the camera(s) can either be mounted on the robot's end-effector, this is known as eye-in-hand configuration, or fixed in the workspace, also known as eye-to-hand [30], [15]. There also exist configurations where, for instance, the camera is mounted on another robot [39] or on a pan/tilt head [25], [12], [13], [11], [14].

2.2.2 Control Architecture

When the visual controller entirely replaces the robot's joints controller and thereby computes low-level joints inputs, the visual servoing architecture is said to be "*direct*" [30], [31]. Implementing such an architecture requires a high speed camera [40, 41][42]. However, when in a hierarchical manner, the visual controller computes set-points for the low level controller (joint's controller) which internally stabilizes the robot, visual servoing is said to be "*indirect*". This control architecture is characterized by two nested loops running at different frequencies (about 25 Hz for the camera and $>$ 100 Hz for the joint's servo control). Its implementation is simple, scalable and robust, with the possibility to separate the robot's kinematic singularities from those of the vision system [43].

2.2.3 Error/Task Function

Classifying visual servoing systems based on the design of features used in the definition of the task function yields the two archetypal approaches namely: *Position Based Visual Servoing* (PBVS) and *Image Based Visual Servoing* (IBVS) [37], [30], [3].

In PBVS, also known as 3D Visual Servoing, the task function is defined by 3D parameters, for instance, the relative pose between the camera and the target, estimated by processing further the

image measurements. However, in IBVS, also known as 2D Visual Servoing, the task function is directly defined in the image space using the 2D image measurements without any 3D estimation.

Note that, there also exist hybrid approaches which combine some characteristics of both PBVS and IBVS, for example the 2D $\frac{1}{2}$ Visual Servoing proposed in [43, 44].

2.3 Image Formation

As can be seen in Figure 2.1, the visual sensor plays a very important role in the control scheme. Indeed, it is used for "measuring" the task execution from the observed images and thereby provides the necessary information for possible corrections. Modeling the image formation process is therefore necessary to establish the relationship between the camera, the observed object and the resulting image.

Assuming that the observed scene can be modeled by geometric entities [36], we present next a geometrical model of image formation using a *pinhole* camera model (see, for example, ref. [45, 46, 47] for other camera models).

2.3.1 Image Formation: Camera model

Consider the *pinhole* camera model based on perspective projection. In this model commonly used [48, p.143] and illustrated in Figure 2.2, it is assumed that all the light rays coming from the scene pass through a single point called *optical center* before reaching the sensor (photosensitive film or CCD/CMOS chip in digital camera). Thus, the image of any point of the scene is formed at the intersection of a projection plane or the *image plane* with the straight line modeling the light ray coming from that point.

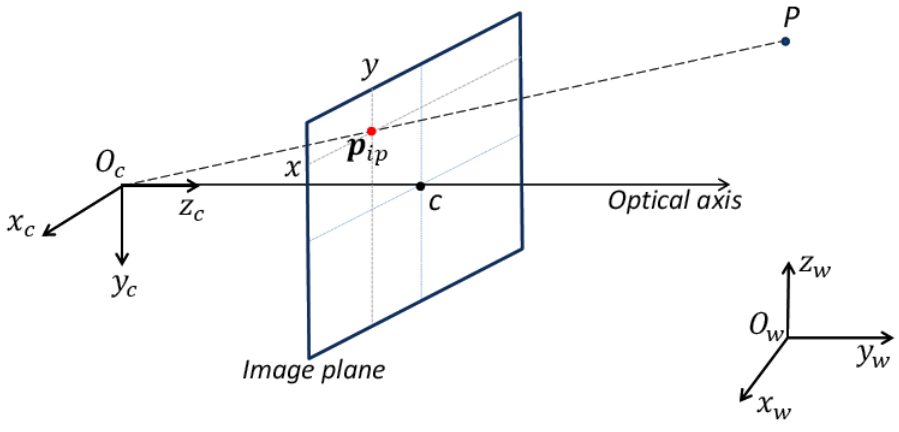


Figure 2.2: Pinhole camera model with perspective projection

Consider a frame $\mathcal{F}_c : \{O_c, x_c, y_c, z_c\}$ associated with the camera and having its origin O_c at the

optical center, and the z_w axis coinciding with the optical axis. Located at a distance f called *focal length*, from the origin O_c, the image plane intersects perpendicularly the optical axis at a point c, called *principal point*. This plane is endowed with a $2D$ reference frame whose axes are respectively parallel to x_c and y_c.

Consider also in a 3D space related to the scene a point P whose coordinates $P_w = (X_w, Y_w, Z_w)$ are expressed relative to the world reference frame $\mathcal{F}_w : \{O_w, x_w, y_w, z_w\}$. Obtaining the image of P in the sensor frame necessitates a series of transformations from the world frame, where P is expressed, to the sensor frame, where the image is expressed, via the camera's frame.

2.3.1.1 Transformation from World to Camera Frame

This is a rigid transformation and it is uniquely determined by a rotation and a translation in $\mathbb{R}^3$ space. Let us denote by ${}^c\mathbf{R}_w$ and ${}^c\mathbf{t}_w$ respectively, the rotation matrix and the translation vector of the world frame with respect to the camera frame. Using homogenous coordinates, the relationship between the coordinates of P expressed in $\mathcal{F}_w$, $P_w = (X_w, Y_w, Z_w, 1)$, and those of the same point P expressed with respect to $\mathcal{F}_c$, $P_c = (X_c, Y_c, Z_c, 1)$, is given by

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} {}^c\mathbf{R}_w & {}^c\mathbf{t}_w \\ 0_{1\times 3} & 1 \end{bmatrix}}_{{}^c\mathbf{M}_w} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.1}$$

The matrix ${}^c\mathbf{M}_w$ is the homogenous transformation from F_w to F_c; it is defined by six parameters (three for rotation and three for translation) called *extrinsic camera parameters* [48, p.143]. More compactly, Equation (2.1) can be written as

$$\boldsymbol{P}_c = {}^c\mathbf{M}_w \boldsymbol{P}_w \tag{2.2}$$

2.3.1.2 Transformation from Camera to Image Space

This transformation is a perspective projection. With reference to Figure 2.2, the point P_c, expressed in $\mathcal{F}_c$, projects onto the image plane in a point $\mathbf{p}_{ip}$ of coordinates (x_{ip}, y_{ip}) such that

$$x_{ip} = f\frac{X_c}{Z_c} \quad , \quad y_{ip} = f\frac{Y_c}{Z_c} \quad \text{and} \quad z_{ip} = f \tag{2.3}$$

By dividing $\mathbf{p}_{ip}$ by f, it can be defined homogenous normalized coordinates $\bar{\mathbf{p}} = (x, y, 1)$ such that

$$\bar{\mathbf{p}} = \frac{1}{Z_c}\mathbf{A}\boldsymbol{P}_c \tag{2.4}$$

where $\mathbf{A} = \begin{bmatrix} \mathbf{I}_{3\times 3} & \mathbf{0} \end{bmatrix}$ and $\mathbf{I}_{3\times 3}$ is a 3×3 identity matrix.

2.3.1.3 Transformation from Image to Sensor Space

In a digital camera, the sensor is an array of photosensitive elements: the pixels. All coordinates (x, y) on image plane have to be expressed in coordinates (u, v) in pixel units. Since the pixel positions are stored in computer as unsigned integers, the origin of the sensor frame is at a corner (e.g. top left) and therefore does not coincide with the principal point (see Figure 2.3). Moreover, the pixel shape is not usually equilateral; and the sensor frame may be skewed such that the angle between the two axes is not equal to 90 degrees. Hence, it can be shown that the pixel coordinates (u, v) are related to the image plane coordinates (x, y) by

$$\begin{cases} u &= fk_u\, x - fk_u \cot\theta\, y + u_0 \\ v &= \frac{fk_v}{\sin\theta}\, y + v_0 \end{cases} \tag{2.5}$$

where k_u and k_v, expressed in pixels/meters, are scaling factors in the u and v direction, respectively. u_0 and v_0 are the pixels coordinates of the principal point and θ is the angle between sensor axes.

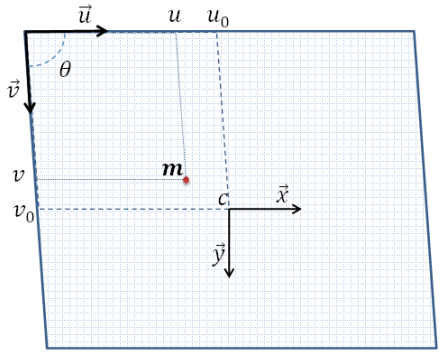


Figure 2.3: Image transformation from metric to pixel unit

Using homogenous pixel coordinates $\boldsymbol{m} = \begin{bmatrix} u & v & 1 \end{bmatrix}^T$, Equation (2.5) can be written as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} fk_u & -fk_u \cot\theta & u_0 \\ 0 & \frac{fk_v}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{K}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.6}$$

The parameters in the $\boldsymbol{K}$ matrix are called *intrinsic camera parameters* [48]. Finally, from Equations (2.2), (2.4) and (2.6), it can be shown that the pixel coordinates of a point P expressed in the world frame will be given by

$$\boldsymbol{m} = \frac{1}{Z_c} \boldsymbol{K}\, \mathbf{A}^c \mathbf{M}_w \boldsymbol{P}_w \tag{2.7}$$

Note that we have assumed in modeling that phenomenon such as distortion in image is negligible,

otherwise it should have been accounted for in the model [49]. Moreover, the transformation from metric units to pixel units requires the calibration of the camera. The calibration problem, which consists of determining intrinsic as well as extrinsic camera parameters, has been widely addressed in literature. One can refer for example to [50], [51].

2.4 Visual Servoing: Formulation

2.4.1 Task Function-based Formulation

The visual servoing problem is usually formulated and solved in terms of regulation to zero a task function by appropriate camera/robot motion [3]. This task function represents the robotic task to be performed, which is defined in terms of desired image features, and constitutes the goal to reach from a given initial configuration [36].

Let us consider the following task function

$$\boldsymbol{e}(t) = \boldsymbol{\xi}\left(r(t)\right) - \boldsymbol{\xi}^{\star} \tag{2.8}$$

where $\boldsymbol{\xi}\left(r(t)\right)$ represents a vector of k selected visual features, $\boldsymbol{\xi}^{\star}$ represents the desired value of $\boldsymbol{\xi}\left(r(t)\right)$ and $r(t) \in SE(3)$ (Special Euclidean space) is a configuration vector, here between the scene and the camera at instant t.

The time derivative of the task function is given by [15]

$$\dot{\boldsymbol{e}}(t) = \frac{\partial \boldsymbol{e}}{\partial \xi}\dot{\boldsymbol{\xi}} + \frac{\partial \boldsymbol{e}}{\partial t} = C\dot{\boldsymbol{\xi}} + \frac{\partial \boldsymbol{e}}{\partial t} \tag{2.9}$$

where $C \in \mathbb{R}^{n\times k}$ is a combination matrix of full rank n mapping the sensor space to the task space, which can be $SE(3)$ or one of its subsets ($n \leq 6$). With the assumption that visual features are defined from geometrical entities, their variations could be related to the motion between the camera/robot and the scene [36]. That is to say

$$\dot{\boldsymbol{\xi}} = \frac{\partial \boldsymbol{\xi}}{\partial r_c}\dot{r}_c + \frac{\partial \boldsymbol{\xi}}{\partial t} = \mathbf{L}_{\xi}\, {}^{c}V + \frac{\partial \boldsymbol{\xi}}{\partial t} \tag{2.10}$$

where $\mathbf{L}_{\xi} \in \mathbb{R}^{k\times 6}$ is the *interaction matrix* associated with the feature $\boldsymbol{\xi}$ [30]. ${}^{c}V = (\upsilon, \omega) \in \mathbb{R}^6$ is the kinematic screw of the camera relative to the object and expressed in the camera frame. With V_c and V_o denoting respectively the induced velocity components due to camera and object motions, we have ${}^{c}V = V_c - V_o$. $\frac{\partial \xi}{\partial t}$ is the variation of $\boldsymbol{\xi}$ due to unknown motion.

When the camera has less than six degrees of freedom or for a redundant robot, it is more convenient to express $\dot{\boldsymbol{\xi}}$ with joint space variables [15], [52]. Thus, Equation (2.10) becomes

$$\dot{\boldsymbol{\xi}} = \frac{\partial \boldsymbol{\xi}}{\partial r_c}\frac{\partial r_c}{\partial r_e}\frac{\partial r_e}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial \boldsymbol{\xi}}{\partial t} = \boldsymbol{L}_\xi \, {}^cW_e \, {}^e\mathbf{J}_q \dot{\mathbf{q}} + \frac{\partial \boldsymbol{\xi}}{\partial t} \tag{2.11}$$

where the subscripts $[.]_c$ and $[.]_e$ of the vector r stand for camera and end-effector, respectively. $\mathbf{q}$ is the joints vector, ${}^e\mathbf{J}_q = \frac{\partial r_e}{\partial \mathbf{q}}$ is the robot Jacobian and ${}^cW_e = \frac{\partial r_c}{\partial r_e}$ is the velocity twist transformation matrix between the end-effector and the camera frame [3]. It is given by

$${}^cW_e \triangleq \begin{bmatrix} {}^c\mathbf{R}_e & [{}^c\boldsymbol{t}_e]_\times \, {}^c\mathbf{R}_e \\ \mathbf{0}_{3\times 3} & {}^c\mathbf{R}_e \end{bmatrix} \tag{2.12}$$

where ${}^c\mathbf{R}_e$ and ${}^c\boldsymbol{t}_e$ are the rotation matrix and translation vector from the end-effector frame to the camera frame and $[{}^c\boldsymbol{t}_e]_\times$ is the skew symmetric matrix associated with ${}^c\boldsymbol{t}_e$. The matrix ${}^cW_e$ can either be constant or can vary depending on the camera-robot configuration.

Using Equations (2.10)-(2.12), the variation of the task function (2.9) can be related to the camera velocity and joints velocity as follows [15]

$$\begin{cases} \dot{e}(t) &= \mathbf{L}_e \, {}^cV + \frac{\partial \boldsymbol{e}}{\partial t} \\ \dot{e}(t) &= \mathbf{J}_e \dot{\mathbf{q}} + \frac{\partial \boldsymbol{e}}{\partial t} \end{cases} \tag{2.13}$$

where $\mathbf{L}_e \triangleq \frac{\partial e}{\partial \xi}\mathbf{L}_\xi$ and $\mathbf{J}_e \triangleq \frac{\partial \boldsymbol{e}}{\partial \xi}\mathbf{J}_\xi$ are, respectively, the interaction matrix and the Jacobian of the task function, while $\mathbf{J}_\xi \triangleq \mathbf{L}_\xi \, {}^cW_e \, {}^e\mathbf{J}_q$ and $\frac{\partial \boldsymbol{e}}{\partial t}$ is the variation of $\boldsymbol{e}(t)$ attributed to object motion.

2.4.2 Designing Visual Control Law

In visual servoing, the control design problem is to find the control input to the robot that will zero the task function.

Assuming that velocity inputs are available on the robot, it is common to choose as control input the velocity [3]. The latter is usually designed to induce an exponential decrease of the task function ($\dot{\boldsymbol{e}}(t) = -\Lambda \boldsymbol{e}(t)$). Thus the control law is

$$\begin{cases} {}^cV &= -\Lambda \, \widehat{\mathbf{L}_e^\dagger} \, \boldsymbol{e}(t) \\ \dot{\mathbf{q}} &= -\Lambda \, \widehat{\mathbf{J}_e^\dagger} \, \boldsymbol{e}(t) \end{cases} \tag{2.14}$$

where the object was assumed to be motionless ($\frac{\partial \boldsymbol{e}}{\partial t} = 0$) and Λ is a gain matrix. $\mathbf{L}_e^\dagger$ is the Moore-Penrose pseudo inverse of $\mathbf{L}_e$. In practice, only its approximation or estimate denoted by the " ˆ " symbol can be known [36], [15].
Apart from $\widehat{\mathbf{L}_e} = \widehat{\mathbf{L}_e}(t)$, computed at each iteration, different choices of $\widehat{\mathbf{L}_e}$ have been reported. For example, a constant matrix at the desired position, denoted $\widehat{\mathbf{L}_{e^*}}$, with the benefit of simplifying the computation [36] , or an average between the current and the desired matrix, $\widehat{\mathbf{L}_e} = \frac{1}{2}(\widehat{\mathbf{L}_e}(t) + \widehat{\mathbf{L}_{e^*}})$ in [53] or even a weighted sum $\widehat{\mathbf{L}_e} = \rho\widehat{\mathbf{L}_{e^*}} + (1-\rho)\widehat{\mathbf{L}_e}(t)$ in [54], with ρ the weighting factor.

2.4.2.1 Stability and Convergence Analysis

To analyze the stability and convergence of a visual servoing system, one can consider the Lyapunov function $\mathcal{L} = \frac{1}{2} e^T e$ [3]. The system is said to be asymptotically stable if it can be proven that $\dot{\mathcal{L}}$ is negative definite. Hence, we have

$$\begin{aligned} \dot{\mathcal{L}} &= e^T \dot{e} \\ &= e^T \mathbf{L}_e \, {}^c V \end{aligned} \tag{2.15}$$

Substituting the control law (2.14) in (2.15) gives

$$\dot{\mathcal{L}} = -\Lambda \, e^T \mathbf{L}_e \widehat{\mathbf{L}_e^{\dagger}} \, e(t) \tag{2.16}$$

Thus, the stability is subject to the positive-definiteness of $(\mathbf{L}_e \widehat{\mathbf{L}_e^{\dagger}}) \in \mathbb{R}^{k \times k}$. If $k > 6$, then there exist local minima, since $rank(\mathbf{L}_e \widehat{\mathbf{L}_e^{\dagger}}) \leq 6$. In that case, one can only conclude on local asymptotic stability [3]. Moreover, the performances of the resulting closed loop system will depend upon the choice of $\boldsymbol{\xi}$ used to define the task function and its associated interaction matrix $\mathbf{L}_{\xi}$.

2.5 Visual Servoing Techniques

Consider a typical visual servoing problem, where a camera supposed to be mounted on a robot observes an $L-$shape object from a given configuration and has to move to a desired one. This is illustrated in Figure 2.4.

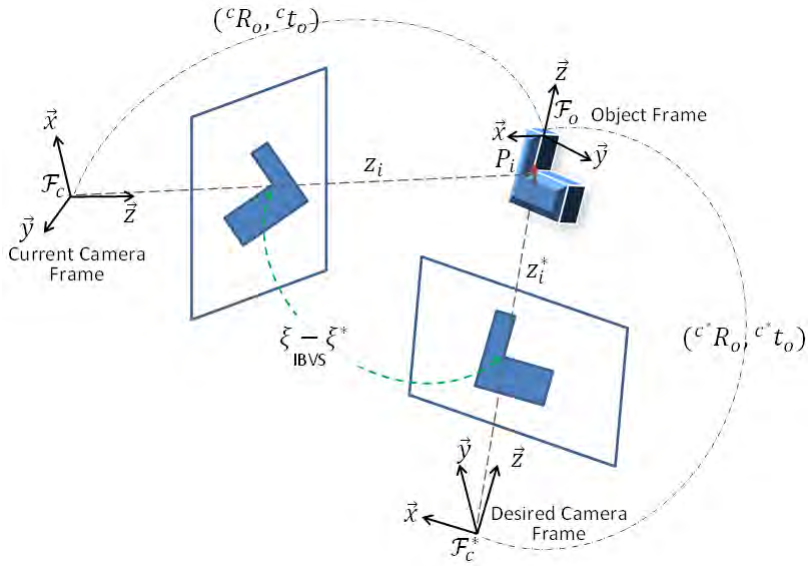


Figure 2.4: Modeling a visual servoing problem

Let $\mathcal{F}_o$ be the reference frame attached to the target object, $\mathcal{F}_c$ and $\mathcal{F}_c^*$ be the reference frames associated with the camera, respectively at the current and desired pose. As mentioned before,

the task function can be defined either in 3D Cartesian space (PBVS) or in 2D image space (IBVS) or even in both spaces (hybrid VS).

2.5.1 Position Based Visual Servoing

In this approach, visual measurements are further processed to estimate the relative pose between the camera/robot and the target object (($^{c}R_{o}$,$^{c}t_{o}$) and ($^{c*}R_{o}$,$^{c*}t_{o}$)) as shown in Figure 2.4. This 3D information is used to define a task function allowing to control the robot in Cartesian space.

Thus, using the computed pose, the features vector $\boldsymbol{\xi}$ can have the following form

$$\boldsymbol{\xi} \triangleq (^{j}\mathbf{t}_{i}, \theta\mathbf{u}) \tag{2.17}$$

where $^{j}\mathbf{t}_{i}$ denotes the translation vector of i with respect to j. $\theta\mathbf{u}$ is the angle/axis parametrization of the rotation, with θ and $\mathbf{u}$ the angle and the axis, respectively.

$\theta\mathbf{u}$ generally chosen equal to $^{c*}\mathbf{R}_{c}$, the rotation between the current and the desired camera frame, becomes *zero* while the camera reaches the desired pose. Regarding $^{j}\mathbf{t}_{i}$, different options have been proposed: If for example, $^{j}\mathbf{t}_{i} = {}^{c*}\mathbf{t}_{c}$, whose desired value is *zero*, $\boldsymbol{\xi}$ is defined as $\boldsymbol{\xi} \triangleq (^{c*}\mathbf{t}_{c}, \theta\mathbf{u})$ [55]. The associated interaction matrix is given by [3]

$$\mathbf{L}_{\xi} = \begin{bmatrix} {}^{c*}\mathbf{R}_{c} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{L}_{\theta\mathbf{u}} \end{bmatrix} \tag{2.18}$$

where $\mathbf{0}_{3\times3}$ is a 3×3 zero matrix, $\mathbf{L}_{\theta\mathbf{u}}$ is a matrix given in [44] by

$$\mathbf{L}_{\theta\mathbf{u}} = \mathbf{I}_{3\times3} - \frac{\theta}{2}[\mathbf{u}]_{\times} + \left(1 - \frac{sinc\theta}{sinc^{2}\frac{\theta}{2}}\right)[\mathbf{u}]_{\times}^{2} \tag{2.19}$$

where the function *sinc* is defined such that $\theta sinc\theta = sin\theta$.

The block diagonal form of the resulting interaction matrix allows a decoupled control between translation and rotation. This gives good 3D trajectory of the camera. However, there is no control in the image and the target object may even leave the field of view. Consequently, the servoing will stop [56].

The translation can also be selected as $^{j}\mathbf{t}_{i} = {}^{c}\mathbf{t}_{o}$, with its desired value $^{c*}\mathbf{t}_{o}$. Hence, $\boldsymbol{\xi}$ would be defined as $\boldsymbol{\xi} \triangleq (^{c*}\mathbf{t}_{o}, \theta\mathbf{u})$ [57], which yields the following interaction matrix

$$\mathbf{L}_{\xi} = \begin{bmatrix} -\mathbf{I}_{3\times3} & [{}^{c}\mathbf{t}_{o}]_{\times} \\ \mathbf{0}_{3\times3} & \mathbf{L}_{\theta\mathbf{u}} \end{bmatrix} \tag{2.20}$$

where $[{}^{c}\mathbf{t}_{o}]_{\times}$is the skew symmetric matrix related to $^{c}\mathbf{t}_{o}$.

The interaction matrix now has a triangular form, which still allows a partial decoupling. Moreover, the control of at least one point of the object (in this case the origin) improves the image trajectories [3] and it also increases the probability of keeping the object in the field of view [43]. However, the 3D trajectory is no longer a straight line as in the first case. PBVS can be summarized by the block diagram shown in Figure 2.5.

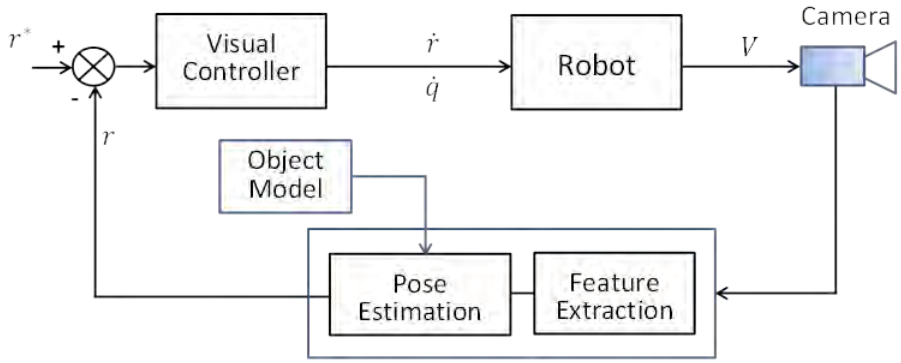


Figure 2.5: Block Diagram of a Typical Position Based Visual Servoing Scheme

In PBVS, the pose estimation requires a well calibrated camera and additional information to the two-dimensional image measurements, such as the object model or the distance camera-object in order to recover the 3D parameters. For instance, image of known object points were used in [58, 59, 60, 61, 62], point to line, respectively, point to region correspondence in [63] and [64], etc. Epipolar geometry was also used to estimate the partial pose, for example in [65, 45, 47] and in [66] with the essential and homography matrix, respectively. Errors in calibration result, however, in 3D reconstruction errors, which drastically affect the way the system converges. With a perfect estimation, PBVS can be globally asymptotically stable [67, 66].

2.5.2 Image Based Visual Servoing

In IBVS, visual information is immediately used to define a task function in the 2D image space, without any 3D reconstruction, in order to control a robotic system [30, 3]. In this approach, the pose estimation is implicitly solved, since a matching between the current and desired image of the object implies that the camera is at the desired pose [68]. Such an approach is illustrated in Figure 2.4 with the green dashed line.

The behavior of the closed loop system will depend on the defined task function, which itself depends upon the choice of image features parameters [69, 56], their number and their configuration [70]. Indeed, there are different possible choices of image features, for example image point, lines [71, 36], moments [72, 73], etc.

We consider here the simplest choice: the image point, whose coordinates will be used to define the task function.

Let $P_i = (X_i, Y_i, Z_i)$ be an object point expressed in the camera frame $\mathcal{F}_c$ and $p_i = (x_i, y_i)$ its projection onto the image plane (see Figure 2.4). If the feature vector is defined as $\xi_i = [\ x_i \quad y_i\ ]^T$, its time derivative could be written as

$$\dot{\xi}_i = \begin{bmatrix} \frac{1}{Z_i} & 0 & -\frac{x_i}{Z_i} \\ 0 & \frac{1}{Z_i} & -\frac{y_i}{Z_i} \end{bmatrix} \dot{P}_i \tag{2.21}$$

with $\dot{P}_i = -v + [P_i]_\times \omega = [-\mathbf{I}_3 \quad [P_i]_\times]^c V$, where $[P_i]_\times$ is the skew symmetric matrix associated with the vector P_i. Hence, substituting the expression of $\dot{P}_i$ in (2.21), allows to relate the features variations to the velocity of the camera as $\dot{\xi}_i = \mathbf{L}_{\xi_i}(x_i, y_i, Z_i)^c V$, where $\mathbf{L}_{\xi_i}(x_i, y_i, Z_i)$ is the associated interaction matrix and it is given by [2, 36]

$$\mathbf{L}_{\xi_i}(x_i, y_i, Z_i) = \begin{bmatrix} -\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_i y_i & -(1+x_i^2) & y_i \\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & (1+y_i^2) & -x_i y_i & -x_i \end{bmatrix} \tag{2.22}$$

Note that the interaction matrix $\mathbf{L}_{\xi_i}$, in addition to the image plane coordinates (x_i, y_i), depends on a 3D information: the depth Z_i of the point, which is usually unknown and has to be estimated. For example with structure from motion techniques as in [74], using nonlinear observer [75, 76].

The metric coordinates (x_i, y_i) can be related, using intrinsic camera parameters, to the actual image measurements in pixel units as follows [43].

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_u} & -\frac{\alpha_{uv}}{\alpha_u \alpha_v} \\ 0 & \frac{1}{\alpha_v} \end{bmatrix} \begin{bmatrix} u_i - u_0 \\ v_i - v_0 \end{bmatrix} \tag{2.23}$$

where $\alpha_u = f k_u$, $\alpha_v = \frac{f k_u}{\sin\theta}$ and $\alpha_{uv} = f k_u \cot\theta$.

By choosing a single feature point, one can only control up to two DoFs. If a task has to be performed in $SE(3)$, six DoFs would be required, which can be theoretically achieved with three feature points. In practice, however, at least four points are necessary. Indeed, it was shown that with three points, there exist four global minima that are impossible to distinguish between [56]. Moreover, the interaction matrix can become singular in some configurations [77], which implies a loss of rank, consequently less controllable DoFs.

Hence, if n feature points are chosen such that $\boldsymbol{\xi} = [\ \xi_1^T \quad \xi_2^T \quad \cdots \quad \xi_n^T\ ]^T$ with $\boldsymbol{\xi} \in \mathbb{R}^{2n}$ and $\xi_i = [\ x_i \quad y_i\ ]^T$, the associated interaction matrix will be given by

$$\mathbf{L}_{\boldsymbol{\xi}} = \begin{bmatrix} \mathbf{L}_{\xi_1} \\ \mathbf{L}_{\xi_2} \\ \vdots \\ \mathbf{L}_{\xi_n} \end{bmatrix} \tag{2.24}$$

Depending on the number of points used, some features may become redundant, which improve robustness to measurement errors and noise. Furthermore, using a "teaching by showing" [31, 3, 78] method to defined the desired features, the positioning accuracy could be made independent from camera calibration and modeling errors [79, 30, 80, 81].

Now using (2.24), the kinematic screw of the camera can be computed from (2.14). However, it could be observed in (2.22) non-linearities and coupling between translational and rotational motions. Additionally, the systems will admit local minima for all error vector $\boldsymbol{e}(t)$ belonging to the null-space of $\widehat{\mathbf{L}_e^{\dagger}}$ [56]. That is

$$\forall\, \boldsymbol{e}(t) \neq 0 \,\&\, \boldsymbol{e}(t) \in \mathbf{Ker}(\widehat{\mathbf{L}_e^{\dagger}}),\ {}^{c}V = 0$$

Moreover, the control being performed on features in 2D image space, good features trajectories could be achieved, thus allowing to keep the object in the field of view. However, there is no control in 3D space, which may result in suboptimal or even unrealizable 3D trajectories, especially for large displacements. With all being said, only local asymptotic stability can be concluded about IBVS [36, 56]. A representation of a typical IBVS control scheme is given in Figure 2.6.

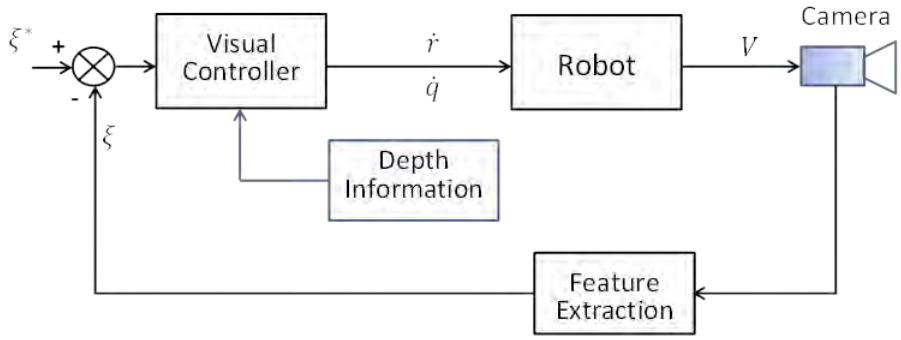


Figure 2.6: Block diagram of a typical Image Based Visual Servoing scheme

Some issues related to IBVS can be resolved by a good choice of visual features. For instance, when the latter are as more representative as possible of the 3D task. A problem such as the backward or forward motion along the optical axis while performing a large rotation about the same axis [56] was handled by shifting features in cylindrical coordinates in [82, 83]. The non-linearity and coupling problem was recently addressed using transformation-invariant features, with image moments, for example in [72, 73], with spherical and unified projection model in [84, 85, 86], or with distances between points in [87].

2.5.3 Other Approaches in Visual Servoing

In response to drawbacks associated with PBVS and IBVS, other visual servoing approaches have been proposed. For instance, hybrid methods particularly 2D 1/2 visual servoing, where the homography between the current and the desired image is used to reconstruct a partial pose

while the remaining information uses IBVS [44], [88], [89]. This offers the benefit of extending the convergence domain of classical IBVS, with good 3D trajectories while keeping the target in the field of view.

Partitioned methods, which try to control separately in a visual servoing scheme some DoFs (e.g. velocities about and along the optical axis or translational velocities) from the remaining DoFs, have also been proposed [90, 91], [88]. Switching schemes between PBVS and IBVS were suggested in [92, 93, 94], [95] to address the visibility and joints limits avoidance problems, or in [96] to avoid local minima and image singularities. Optical flow based method such as d2D/dt visual servoing was reported in [97] for a navigation task.

For large camera displacements and to handle additional constraints, the coupling of visual servoing with path planning methods was also suggested in [98] with potential fields [99], in [100, 101] via navigation functions, in [102, 103, 104] with interpolation in the projective space, or in [105] with homogeneous form and linear matrix inequality (LMI) optimization. We can also mentioned the fusion of visual and kinematic information with an Iterative Extended Kalman Filter in [14]; the purpose was to keep servoing even if the object is out of the field of view.

After introducing fundamental concepts of visual servoing and presenting the two archetypal approaches, in the next section we present and discuss some reported applications of visual servoing on humanoid robots.

2.6 Visual Servoing and Humanoid Robots

As mentioned in the introduction, we will distinguish the reported applications of visual servoing on humanoid robot from a perspective of the performed tasks. Considering whether the task involves the upper body parts alone, the lower body part alone or the whole body.

2.6.1 Visual Servoing Based Humanoid Upper Body Tasks

By visual based upper body tasks, we mean all tasks involving the humanoid's head, the two arms and the torso when it is endowed with joint(s). These tasks are for instance, gazing at, reaching and grasping an object or manipulating an object under visual feedback. A typical control scheme of such applications is illustrated in Figure 2.7.

For grasping and manipulation tasks, the vision system is often used to estimate the 3D relative pose between the robot's hand or gripper and the object. The reconstructed pose can be exploited directly with an already existing controller on the robot, since the 3D information has the same nature as those provided by other type of sensors. However, for robustness reasons, this information is exploited in closed loop, which yields nothing but PBVS.

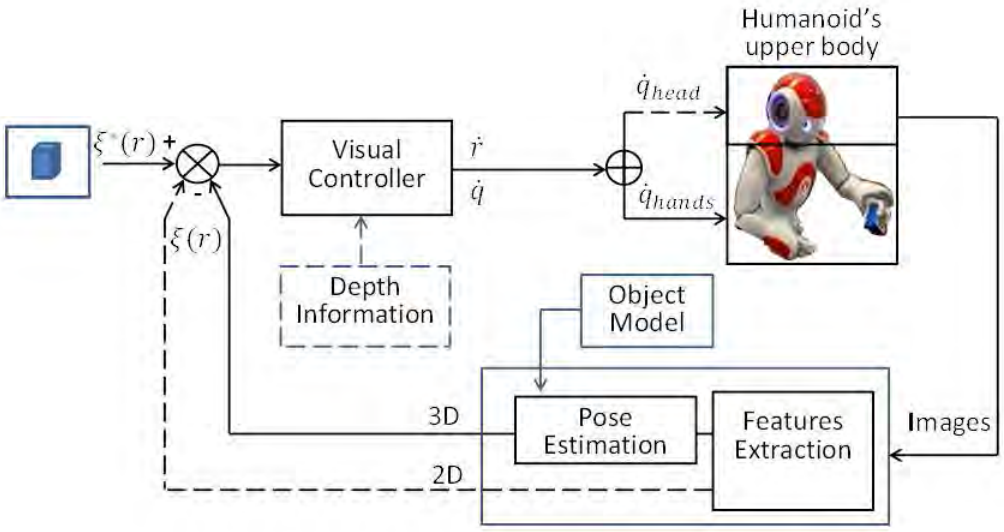


Figure 2.7: Block diagram of Visual servoing of a humanoid upper body

Thus, the required pose can be estimated for instance with multiple cameras, especially stereo-vision when the robot is equipped with such cameras. For example, stereo-vision is exploited in [4] in conjunction with a model based online calibration technique using Kalman filter. For a reach-to-grasp task, stereo-vision is also used in [5], where an open loop approach is used for the reaching and PBVS for grasping an object with humanoid robot BHR-02. *Hubner et al.*, [6] used a similar approach with humanoid ARMAR III in a kitchen environment, where a 3D shape approximation technique based on box primitives [106] is exploited in order to generate grasp hypotheses to be executed by visual servoing.

The pose had also been estimated with a single camera, but using additional information such as prior knowledge of the object model, or with other sensor measurements. For example in [10] and [13], *Moughlbay et al.* used model based visual servoing for grasping and manipulation with humanoid NAO. In a human-humanoid robot (HPR-2) cooperative task, which consisted of transporting a beam while keeping it horizontal, a model based visual servoing is used in [107] to generate reference trajectories for an impedance controller [108] regulating the interactions forces. In [11], augmented reality markers are exploited to estimate the hand and object's poses for manipulation task with REEM Humanoid robot.

Visual and other sensors measurements had also been combined to either enhance the servoing performance or to overcome some drawbacks or limitations related to classical visual servoing schemes. For instance, considering calibration errors and occlusion problems, *Taylor et al.*, [14] fused visual and kinematic measurements in an Iterative Extended Kalman Filter (IEKF), which yielded a hybrid position based visual and kinematic servoing. In [109], a hybrid scheme combining visual, force/torque and kinematic measurements was developed for grasping and manipulation

task with humanoid ARMAR III. The gripper and the object's positions and orientations are respectively derived from visual and kinematic measurements, while the force/torque sensor is used to detect contacts and collisions.

Some grasping and manipulation tasks have also been implemented without pose estimation. For instance, an IBVS technique (Fixation Point Servoing [110]) was used in [12] to map image coordinates to actuators commands thanks to a neural network. A linear approximation between the binocular space and the arm joints space, called "Linear Visual Servoing (LVS)" was proposed in [7], and also used in [8] and [9].

In the above works, whenever it had been performed, the gazing task was used to ensure visibility of both the target and the humanoid's hand or gripper. Often it exploits IBVS with a single camera or an active vision system which maximizes visual information.

2.6.2 Visual Servoing Based Humanoid Lower Body Tasks

The visually controlled lower body tasks considered in this section are essentially locomotion tasks (e.g. navigation, positioning, tracking, etc.). These tasks are more challenging when compared to upper body tasks. Indeed, considering a biped locomotion, the feet alternatively step on the ground, exerting only unilateral constraints, while the center of mass moves continuously [111]. As a result, the latter cannot be controlled directly to guarantee a stable gait. This problem is generally addressed by exploiting the robot momentum and dynamics in a gait planner [112], which often requires knowledge beforehand of the next footstep in order to generate a center of mass motion able to maintain the gait stability [113] (see Figure 2.8).

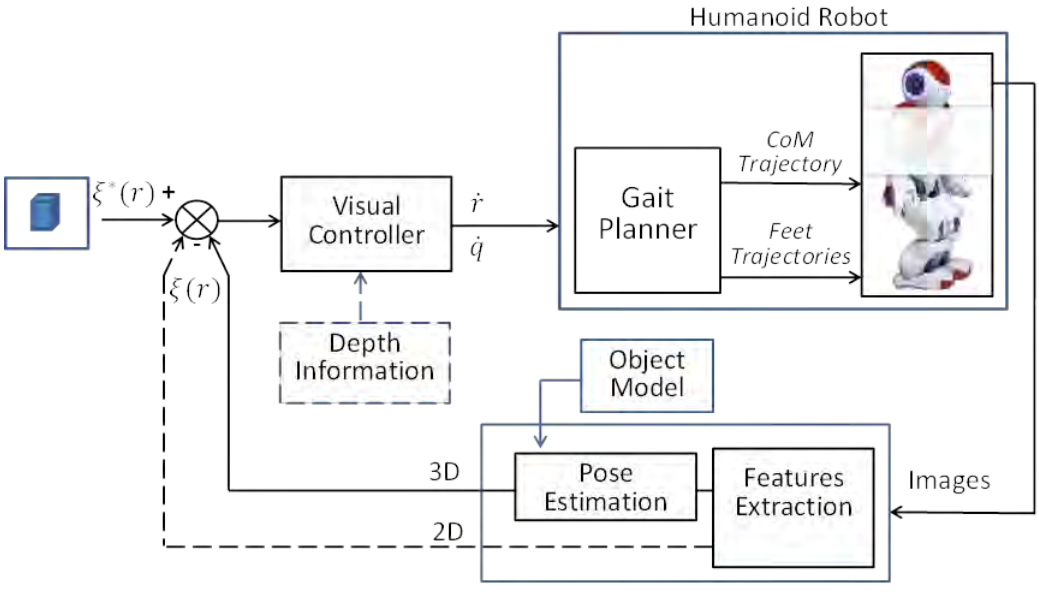


Figure 2.8: Block diagram of Visual servoing of a humanoid Lower body

Thus, the reference commands from the vision system, generally velocities, have to be converted to compatible inputs for the gait planner or can be used directly if the latter offers such an option (e.g. velocity based gait generator).

One of the early studies to have considered this problem is [114]. To drive reactively the locomotion by visual feedback, two translational degrees of freedom were added at the center of the robot. *Lorch et al.*, [115] studied the interaction between vision and biped walking. Stereo-vision is exploited to reconstruct the robot environment (objects localization, poses estimation). The locomotion module uses a step planner, which continuously re-plan the step sequence to take into account possible changes in the perceived environment. *Michel et al.*, [116] combined visual sensing to a footstep planner and achieved autonomous walking towards desired positions while avoiding obstacles. To meet the fast replanning requirement, previous planned steps were reused with only partial recomputation of the current one. An overhead camera was used in the experiments which were carried out on ASIMO humanoid robot.
Asano and Kawamura [117] considered a "Visual tracking Walk" where a humanoid robot with an embedded camera follows autonomously a target based on image feature motion. A discrete version of IBVS with pseudo-Jacobian is used to generate the reference walking trajectory for the step planner. However, a steady state error was noted and it was attributed to the orientation. Considering this problem, it was implemented in [118] a "visual tracking walk" emphasizing this time on the decoupling of the rotational motion in order to stabilize the visual walking. Similarly, this control scheme is applied in [119] to a biped walking robot in a living environment.
Song et al. [120] used PBVS to stabilize the standing and walking of a humanoid robot. In this "visual gyroscope", the pose of an object in front of the robot is estimated using an on-board stereo-vision system. Then, the visual controller exploits this information to generate force/torque commands at the joints in order to maintain a desired orientation of the robot's head. A similar approach was used in [121] in order to control the walking direction by generating torsional deflection at a flexible ankle.
In [13], a model based technique is proposed and implemented on the NAO robot. The visual system is used for objects poses extraction and relative localization of the robot. The reconstructed 3D information is exploited first in open loop for the locomotion task toward the object of interest, and then in closed loop to perform the grasping task. Also with NAO robot, [24] considered a navigation task in a corridor with turns and T-junctions. IBVS is used to generate first velocity commands for a virtual unicycle mobile robot, whose driving velocity is assumed to be constant, and then transformed to reference velocity for a humanoid robot. Two image features, a vanishing point and a middle point, were defined from the image of the floor-walls edges of the corridor.

All the methods described above rely however, on classical footstep planners. Though fast replanning strategies have been proposed if confronted to sudden obstacles, these planners require

knowledge beforehand of at least the next footstep. This implies that there would exist a delay of at least one step period between the reference command and the action. As a consequence, in highly reactive applications, this delay will drastically affect the performances.

Addressing this reactivity issue, *C. Dune et al.*, [19, 20] proposed a visual based dynamic walking, which was achieved thanks to a direct coupling of visual servoing with the online pattern generator developed by *Herdt et al.*, [23, 27]. In this approach, the footsteps are reactively controlled by visual perception with a pattern generator taking directly velocities as inputs. As in [24], the camera was, however, assumed to be rigidly linked to the torso, denying as such all gazing motions.

2.6.3 Visual Servoing Based Humanoid Whole Body Tasks

The visually guided whole body tasks considered here are those involving simultaneously upper body and lower body tasks discussed previously. Combining the two types of tasks brings more challenges, especially for stability, but it also offers significant benefits. Owing to the increased number of DoFs, the workspace is enlarged and the robot becomes redundant with respect to tasks. This could be exploited, for example, to improve visibility, manipulability or to avoid joint limits or even obstacles, etc.

This perception-action approach to generate whole body motion was suggested for highly reactive contexts in [114], where IBVS was used to control a humanoid in a virtual reality framework. Behaviors that could be specified as visual tasks such as gazing at an object, following path, tracking trajectory or target, etc. were considered. Exploiting the redundancy, an approach to avoid joint limits was also proposed. *O. Lorch et al.*, [115] considered the problem of autonomous locomotion behavior on walking machines. To maximize objects visibility while accounting for intermediate goal of the walking machine, a gaze controller was designed. All possible changes in the perceived environment were accounted for at the locomotion level thanks to a continuous replanning strategy of steps sequence.

Within a task sequencing framework, *Mansard* [25] applied visual servoing as a secondary task to grasp a ball while walking. The overall task is divided into a number of subtasks to be executed. During the walk, a gaze control keeps the target at the center of the image, while the hand is visually driven towards the target, and finally grasps the latter when close enough. The experiment was carried out on a HPR-2 humanoid robot.

2.6.4 Controller Design

In the reviewed works, indirect visual servoing is the main control architecture used, except in [120] and [121], where the visual controller compute directly the force/torque to be applied at the joints.

To ensure convergence of the system, most of them have used a simple proportional control law. However, in [12] a neural network was used to map the images coordinates to the motor commands. In [24], where the concept of virtual unicycle was used, a feedback linearization technique was applied in an IBVS scheme to compute the only control variable, the steering velocity.

Additional compensation schemes addressing, for example, the delay due to image processing and the disturbance due to the sway motion were proposed in [117] and [19]. In [11] the interactions between all upper body tasks were considered and also compensated with a feed-forward control action. Moreover, the redundancy was exploited in the controller design to avoid joints limits using a large projector operator [122]. Similarly, in [114] a new joint limits avoidance technique was proposed.

2.7 Conclusion

In this chapter, after introducing the fundamentals of visual servoing and presenting the two main techniques, we have discussed some applications of these techniques on humanoid robots. We have separated these applications in three categories: the servoing, respectively, of the upper body, the lower body and the whole body of a humanoid robot. This classification was based, from the humanoid perspective, on the level of complexity required to perform a given task.

We have seen that the problem of servoing of a humanoid upper body is equivalent to that of a fixed-base robot manipulator. However, due to the biped locomotion of a humanoid, visual servoing of the lower part relies essentially on a footsteps planner to account for different walking constraints. In the servoing of the whole body the two previous cases are combined and their interactions have also to be accounted for. Regarding the vision system, it had been generally used for total or partial pose reconstruction in order to extract necessary information for grasping in case of manipulation, or information for the gait planner whenever locomotion is involved.

In the reviewed works, a systematic and general approach is unfortunately lacking on how to apply visual servoing for walking or for whole body tasks. In some works, extra DoFs are added to the base, in others, the velocity commands are converted to positions for the planner or else, the humanoid is assimilated to a unicycle robot. Nevertheless, some works laid good foundations for whole body reactive control with multiple tasks management, including their interactions as we will see in chapter 4, where a general approach is proposed.

Chapter 3

Modeling and Control of Biped Humanoid Robot

This chapter deals with modeling and locomotion control aspects of a humanoid robot. In modeling, it focuses on the dynamics of a humanoid robot, where the single support model, the double support and the impact models are provided. In the control section, after introducing the concepts related to ZMP control schemes, the predictive control based pattern generation problem is discussed, with a particular emphasis on the pattern generator with automatic footsteps placement [23, 27]. Following this discussion, an extension accounting explicitly for rotational velocity on the automatic footsteps placement scheme is proposed.

3.1 Modeling

As a mechanical structure, a robot comprises a set of rigid bodies, called links, connected together by joints [123]. The joints are generally revolute or prismatic and allow relative motion between two links. They are often actuated so that the overall motion of the robot can be controlled in order to perform a task. A robotic task is usually defined in terms of location of the end-effector (EE) characterized by a position and an orientation. Unlike a fixed robot, a humanoid robot is characterized by a mobile base and multiple end-effectors that can interact with its environment. Its bipedal locomotion can only be realized through contact forces when it steps on the ground.

Modeling a robot refers to deriving a set of mathematical expressions that describe the geometrical and time-based properties of the robot motion [124]. The complete model of a robot mechanism is usually divided in two parts: the robot kinematics and the robot dynamics.

3.1.1 Kinematic Modeling of Humanoid Robot

The kinematics of a humanoid robot allows to determine the relations between its joints configuration and the location of all its end-effectors or any other point on the robot. To better describe

relative locations or poses (position and orientation) between links, it is necessary to assign frames to each link. Hence, the kinematics is known if the changes of these frames locations in relation to the joints are established. However, due to its mobility, the configuration of a humanoid robot will be completely defined with respect to a fixed inertial frame, if in addition to the internal configuration the pose of at least one of its links is specified with respect to that inertial frame.

Let $\mathbf{q}$ be the vector of N_t independent coordinates (generalized coordinates) defining, over an N_t-dimensional manifold $\mathcal{C}$, the configuration of the robot. $\mathbf{q}$ is given by

$$\mathbf{q} = \begin{bmatrix} q_1 & q_2 & \dots & q_{N_t} \end{bmatrix}^T \tag{3.1}$$

Let also define $\boldsymbol{X}_i$ as a vector of n_e *operational coordinates* (see [125]) defining, over an n_e-dimensional manifold $\mathcal{O}$, the end-effector's location of the serial chain i, such that

$$\boldsymbol{X}_i = \begin{bmatrix} \mathcal{X}_{i_1} & \mathcal{X}_{i_1} & \dots & \mathcal{X}_{i_{n_e}} \end{bmatrix}^T \qquad \text{with}\, 0 < n_e \leq 6 \tag{3.2}$$

Using the above definitions, the the kinematics analysis can be carried out starting with the forward kinematics, the inverse kinematics and then the velocity kinematics.

3.1.1.1 Forward Kinematics

In general, the forward kinematics problem is to find the relative pose between any two selected parts of a robot given the robot's geometrical parameters and the values of the joints [123]. In this particular case, we seek to express all end-effectors operational coordinates $\boldsymbol{X}_i$ as function of the generalized coordinates $\mathbf{q}$. This mapping can be represented by a function $\psi : \mathcal{C} \to \mathcal{O}$, such that

$$\mathbf{q} \longmapsto \boldsymbol{X}_i = \psi(\mathbf{q}) \tag{3.3}$$

3.1.1.2 Inverse Kinematics

The inverse kinematics problem is to compute the joints values $\mathbf{q}$ that will yield a given end-effector pose $\boldsymbol{X}_i$. If it exists [124], [126, chap. 3], this mapping can be written as $\psi^{-1} : \psi(\mathcal{C}) \to \mathcal{C}$, such that

$$\boldsymbol{X}_i \longmapsto \mathbf{q} = \psi^{-1}(\boldsymbol{X}_i) \tag{3.4}$$

where ψ^{-1} is one reciprocal function of ψ [127].

3.1.1.3 Velocity Kinematics

The aim of velocity kinematics analysis is to derive the velocity expressions, relating the linear and angular velocities of the end-effector (or any other point on the robot) to the joint velocities.

By defining $V_{iE} \triangleq [\ \boldsymbol{\upsilon}_{iE}^T \quad \boldsymbol{\omega}_{iE}^T\ ]^T$ as the velocity of EE_i, with υ_{iE} and ω_{iE} the linear and angular velocity of EE_i, the velocity relationships are determined as

$$V_{iE} \triangleq \begin{bmatrix} \boldsymbol{\upsilon}_{iE} \\ \boldsymbol{\omega}_{iE} \end{bmatrix} = \mathbf{J}_i(\mathbf{q})\dot{\mathbf{q}} \tag{3.5}$$

where $\mathbf{J}(\mathbf{q})$ is the $n_e \times N_t$ Jacobian matrix of the function ψ defined by the forward kinematics. It is then a mapping between the tangent spaces $\mathbf{J}(\mathbf{q}) : T_{\mathbf{q}}\mathcal{C} \rightarrow T_{\boldsymbol{X}_i}\mathcal{O}$, such that [127], [124]

$$\mathbf{J}_i(\mathbf{q}) = \frac{\partial \psi_i}{\partial \mathbf{q}} \tag{3.6}$$

3.1.2 Dynamic Modeling of Humanoid Robot

The dynamics of a humanoid relates its motions to the forces/torques causing or resulting from these motions [128]. The latter are described by the time evolution of the robot configuration.

Hence, let us define a fixed inertial frame $\mathcal{F}_W = (O_w, \overrightarrow{x}_w, \overrightarrow{y}_w, \overrightarrow{z}_w)$ (world frame) as shown in Figure 3.1, and let the internal configuration of the humanoid robot be parametrized by a vector $\mathbf{q}_h = [\ \ q_{h_1} \quad q_{h_2} \quad \dots \quad q_{h_{N_j}}\ \ ]^T \in \mathbb{R}^{N_j}$, with N_j number of the robot's joints. Thus, in order to describe the global position and orientation of a humanoid, let $\mathcal{F}_b$ be the frame assigned to the torso (taken as body reference frame) and let $\mathbf{q}_b = [\ \mathbf{t}_b^T \quad \boldsymbol{\phi}_b^T\ ]^T \in \mathbb{R}^3 \times SO(3)$ be the pose of $\mathcal{F}_b$ with respect to $\mathcal{F}_W$, where $\mathbf{t}_b$ and $\boldsymbol{\phi}_b$ represent the position and orientation of $\mathcal{F}_b$ relative to $\mathcal{F}_W$, respectively (see Figure 3.1).

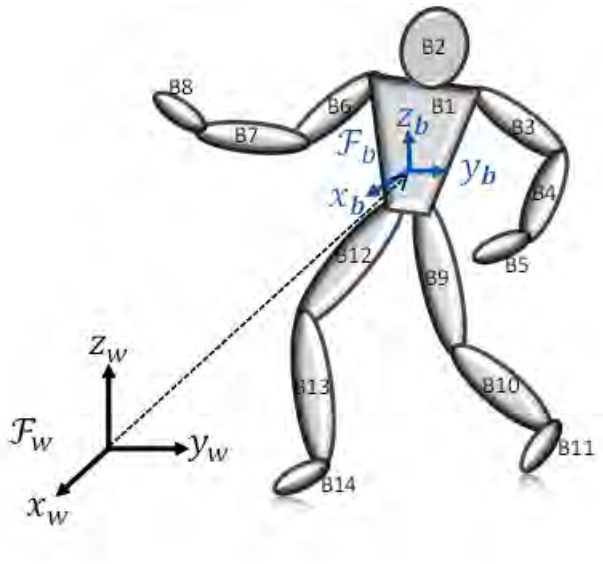


Figure 3.1: Representation of Humanoid NAO links in a fixed inertial frame

The set of generalized coordinates of the robot is then

$$\mathbf{q} = [\ \mathbf{q}_h^T \quad \mathbf{q}_b^T\ ]^T \tag{3.7}$$

where $\mathbf{q} \in \mathbb{R}^{N_t}$, with $N_t = N_j + 6$.

However, during locomotion, characterized by a single support phase (SSP), double support phase (DSP) and transition phase with impacts [16], $\mathbf{q}_h$ and $\mathbf{q}_b$ are no longer independent, their velocities being related by the walking constraints. Therefore, in order to model this constrained system, it is derived first the Lagrange equations for the unconstrained robot (with a free floating base) and then the constraints forces could be accounted for using D'Alambert principle [128].

3.1.2.1 Floating Base Dynamic Model

Using Lagrange formulation, it can be shown that the dynamic model of the unconstrained humanoid robot (with no contact with the ground) can be written as [129]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{\Gamma} + \mathbf{\Gamma}_{ext} \tag{3.8}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{N_t \times N_t}$ is the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{N_t \times N_t}$ is the matrix of centrifugal and Coriolis effects, $\mathbf{G}(\mathbf{q}) \in \mathbb{R}^{N_t}$ is the vector of gravitational forces and $\mathbf{\Gamma} \triangleq [\ \boldsymbol{\tau}_a^T \quad \mathbf{0}_{1\times 6}\ ]^T \in \mathbb{R}^{N_t}$ represents the vector of applied generalized forces, with $\boldsymbol{\tau}_a \in \mathbb{R}^{N_j}$ denoting the vector of actuators forces/torques, while $\mathbf{\Gamma}_{ext}$ represents all external forces/torques acting on the robot.

With reference to Figure 3.1, let us assume that the body mass and inertia properties of the humanoid are concentrated in N_{lk} links ($N_{lk} = 14$ in Figure 3.1). Let us also define $\mathbf{t}_b \triangleq [\ x_b \quad y_b \quad z_b\ ]^T$ and $\boldsymbol{\phi}_b^T \triangleq [\ \alpha \quad \beta \quad \gamma\ ]^T$ which is an Euler angles parametrization of $\boldsymbol{\phi}_b^T \in SO(3)$ such that the generalized coordinates are given by

$$\mathbf{q} = \begin{bmatrix} q_{h_1} & q_{h_2} & \dots & q_{h_{N_j}} & x_b & y_b & z_b & \alpha & \beta & \gamma \end{bmatrix}^T \tag{3.9}$$

The Robot's Inertia Matrix The inertia matrix $\mathbf{M}(\mathbf{q})$ can be derived from the kinetic energy as follows

$$K = \sum_{i=1}^{N_{lk}} \left\{ \frac{1}{2} m_i \boldsymbol{\upsilon}_{gi}^T \boldsymbol{\upsilon}_{gi} + \frac{1}{2} \boldsymbol{\omega}_{gi}^T \mathbf{R}_{gi} \mathbf{I}n_i \mathbf{R}_{gi}^T \boldsymbol{\omega}_{gi} \right\} \tag{3.10}$$

where m_i, $\mathbf{I}n_i$, $\boldsymbol{\upsilon}_{gi}$ and $\boldsymbol{\omega}_{gi}$ denote respectively the mass, the inertia matrix, the linear and the angular velocity of the i^{th} link. $\mathbf{R}_{gi} = \mathbf{R}(\phi_b)\mathbf{R}_i(\mathbf{q}_{hr})$ is the rotation matrix from the i^{th} link's frame to the world frame $\mathcal{F}_W$, while $\mathbf{R}_i(\mathbf{q}_{hr})$ is the rotation matrix from the i^{th} link's frame to the base frame $\mathcal{F}_b$ and $\mathbf{R}(\phi_b)$ the rotation from $\mathcal{F}_b$ to $\mathcal{F}_W$. From the kinematics of the i^{th} link, we can write

$$\boldsymbol{\upsilon}_{gi} = \mathbf{R}(\phi_b)\boldsymbol{\upsilon}_i + \boldsymbol{\upsilon}_b + \boldsymbol{\omega}_b \times \mathbf{R}(\phi_b)\boldsymbol{r}_{ci} \tag{3.11}$$

$$\boldsymbol{\omega}_{gi} = \mathbf{R}(\phi_b)\boldsymbol{\omega}_i + \boldsymbol{\omega}_b \tag{3.12}$$

where $\boldsymbol{\upsilon}_i$ and $\boldsymbol{\omega}_i$ are respectively the linear and angular velocity of i^{th} link's center of mass (CoM) with respect to $\mathcal{F}_b$. $\boldsymbol{\upsilon}_b$ and $\boldsymbol{\omega}_b$ are respectively the linear and angular velocity of $\mathcal{F}_b$ with respect to $\mathcal{F}_W$, and $\boldsymbol{r}_{ci}$ is the position vector of the i^{th} link's CoM expressed in $\mathcal{F}_b$.

From (3.11) and (3.12) we can define the Jacobian matrices $\mathbf{J}_{\upsilon_i}(\mathbf{q})$ and $\mathbf{J}_{\omega_i}(\mathbf{q}) \in \mathbb{R}^{3\times N_j}$ such that $\upsilon_{gi} = \mathbf{J}_{\upsilon_i}(\mathbf{q})\dot{\mathbf{q}}$ and $\omega_{gi} = \mathbf{J}_{\omega_i}(\mathbf{q})\dot{\mathbf{q}}$; they are given by

$$\mathbf{J}_{\upsilon_i}(\mathbf{q}) = \begin{bmatrix} \mathbf{R}(\phi_b)\mathbf{J}_{\upsilon_i}(\mathbf{q}_{hr}) & \mathbf{I}_{3\times 3} & -[\mathbf{R}(\phi_b)\boldsymbol{r}_{ci}]_\times \mathbf{T}(\phi_b) \end{bmatrix} \tag{3.13}$$

$$\mathbf{J}_{\omega_i}(\mathbf{q}) = \begin{bmatrix} \mathbf{R}(\phi_b)\mathbf{J}_{\omega_i}(\mathbf{q}_{hr}) & \mathbf{0}_{3\times 3} & \mathbf{T}(\phi_b) \end{bmatrix} \tag{3.14}$$

with $\mathbf{J}_{hr_i}(\mathbf{q}_{hr}) = \begin{bmatrix} \mathbf{J}_{\upsilon_i}(\mathbf{q}_{hr}) \\ \mathbf{J}_{\omega_i}(\mathbf{q}_{hr}) \end{bmatrix} \in \mathbb{R}^{6\times N_j}$ denoting the Jacobian associated with the center of mass (CoM) of the i^{th} link expressed in $\mathcal{F}_b$, the notation $[\boldsymbol{r}_i]_\times$ denotes the skew symmetric matrix associated with the vector $\boldsymbol{r}_i$, and $\mathbf{T}(\phi_b)$ is a matrix mapping the rotational velocity of $\mathcal{F}_b$ to $\dot{\boldsymbol{\phi}}_b$, such that $\boldsymbol{\omega}_b = \mathbf{T}(\phi_b)\dot{\boldsymbol{\phi}}_b$ [130, 126].

Thus, using Equations (3.13) and (3.14) in (3.10), the kinetic energy K as function of the generalized velocities is given by

$$K = \frac{1}{2}\dot{\mathbf{q}}^T \left[\sum_{i=1}^{N_{lk}} \left\{ m_i \mathbf{J}_{\upsilon_i}^T(q)\mathbf{J}_{\upsilon_i}(q) + \mathbf{J}_{\omega_i}^T(q)\mathbf{R}_{gi}(q)\mathbf{I}n_i\mathbf{R}_{gi}^T(q)\mathbf{J}_{\omega_i}(q) \right\} \right] \dot{\mathbf{q}} \tag{3.15}$$

Hence, from (3.15) we can extract $\mathbf{M}(\mathbf{q})$, which is given by [126]

$$\mathbf{M}(\mathbf{q}) = \sum_{i=1}^{N_{lk}} \left\{ m_i \mathbf{J}_{\upsilon_i}^T(q)\mathbf{J}_{\upsilon_i}(q) + \mathbf{J}_{\omega_i}^T(q)\mathbf{R}_{gi}(q)\mathbf{I}n_i\mathbf{R}_{gi}^T(q)\mathbf{J}_{\omega_i}(q) \right\} \tag{3.16}$$

The Centrifugal and Coriolis Matrix The centrifugal and Coriolis matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ can be derived from the inertial matrix $\mathbf{M}(\mathbf{q})$ using the Christoffel symbol of first type as follows [16]

$$c_{kj}(\mathbf{q}) = \sum_{i=1} \frac{1}{2} \left\{ \frac{\partial M_{kj}(\mathbf{q})}{\partial q_i} + \frac{\partial M_{ki}(\mathbf{q})}{\partial q_j} - \frac{\partial M_{ij}(\mathbf{q})}{\partial q_k} \right\} \dot{q}_i \tag{3.17}$$

where the indices $i, j, k = 1 \dots N_j$.

The Gravity Forces The vector of gravity forces $\mathbf{G}(\mathbf{q})$ is directly derived from the potential energy as follows

$$G_i(\mathbf{q}) = \frac{\partial P(\mathbf{q})}{\partial q_i} \tag{3.18}$$

with the potential energy $P(\mathbf{q})$ given by

$$P(\mathbf{q}) = \sum_{i=1}^{N_{lk}} m_i \, g \, h_{ci}(\mathbf{q}) \tag{3.19}$$

where g is the gravity acceleration and h_{ci} is the height of the i^{th} link's CoM with respect to $\mathcal{F}_W$.

Note that all kinematic variables ($\mathbf{J}_{v_i}(q)$, $\mathbf{J}_{\omega_i}(q)$, $\boldsymbol{r}_{ci}$, $\mathbf{R}_{gi}(q)$, and h_{ci}) are computed from the forward kinematics of the humanoid. In Chapter 5, we will derive the complete kinematic model of humanoid robot NAO, used as experimental platform.

3.1.2.2 Contact Dynamics

Unilateral constraints When at least one of the feet is in contact with the ground (modeled as a rigid body), the contact points of the robot cannot penetrate the ground; therefore their normal distances $\varphi_n(\mathbf{q})$ with respect to the ground are constrained to be greater or equal to zero. Similarly, the resulting ground reaction forces λ_n normal to the contact points can only be positive or zero, given that the ground can only push on the robot foot, but cannot pull, otherwise the "non-penetration" constraint would be violated. These unilateral constraints imply complementary conditions between $\varphi_n(\mathbf{q})$ and λ_n [131]

$$\lambda_n^T \varphi_n(\mathbf{q}) = 0, \quad \text{with} \quad \varphi_n(\mathbf{q}) \geq 0 \quad \text{and} \quad \lambda_n \geq 0 \tag{3.20}$$

which tells us that when there is no contact, $\lambda_n = 0$ and $\varphi_n(\mathbf{q}) > 0$, if the contact is established $\varphi_n(\mathbf{q}) = 0$ then $\lambda_n > 0$. In the latter case ($\varphi_n(\mathbf{q}) = 0$), due to the "non-penetration", it also results constraints on the velocity and acceleration of the system [16], thus

$$R_n(\mathbf{q})\dot{\mathbf{q}} = 0 \tag{3.21}$$

$$R_n(\mathbf{q})\ddot{\mathbf{q}} + s_n(\mathbf{q}, \dot{\mathbf{q}}) \geq 0 \tag{3.22}$$

where $R_n(\mathbf{q}) = \frac{\partial \varphi_n(\mathbf{q})}{\partial q}$ and $s_n(\mathbf{q}, \dot{\mathbf{q}}) = \frac{d}{dt}\left(\frac{\partial \varphi_n(\mathbf{q})}{\partial q}\right)\dot{\mathbf{q}}$.

No Slipping Constraints When a robot in contact with the ground is moving without slipping, the resulting friction forces tangential to the ground oppose any slipping motion. In other words, the "non-slipping" imposes constraints on the contact points displacements tangential to the ground ($\varphi_t(\mathbf{q})$) and on their time derivatives (velocity, acceleration, etc.) [132]. That is

$$\varphi_t(\mathbf{q}) = 0, \; R_t(\mathbf{q})\dot{\mathbf{q}} = 0 \text{ and } R_t(\mathbf{q})\ddot{\mathbf{q}} + s_t(\mathbf{q}, \dot{\mathbf{q}}) = 0 \tag{3.23}$$

where $\varphi_t(\mathbf{q})$ is the vector of contact points displacements parallel to the ground, and $R_t(\mathbf{q})$ and $s_t(\mathbf{q}, \dot{\mathbf{q}})$ are derived as for $\varphi_n(\mathbf{q})$.

Note that the tangential friction forces λ_t are limited and could possibly fail to prevent a slipping

motion. In fact, according to the Amontons-Coulomb's dry friction model [133], when a contact between two objects is subject to a normal force λ_n, the tangential friction force λ_t has its values within the friction cone

$$\|\lambda_{t_i}\| \leq \mu_f \lambda_{n_i} \tag{3.24}$$

where μ_f is the coefficient of friction, mainly dependent on the type of materials in contact.

3.1.2.3 Constrained Dynamical Model

The robot in contact with the ground becomes a constrained dynamical system; the resulting model while accounting for the unilateral and no slipping constraints is given by [16, 129, 111]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{\Gamma} + R_n(\mathbf{q})\lambda_n + R_t(\mathbf{q})\lambda_t \tag{3.25}$$

$$R_n(\mathbf{q})\ddot{\mathbf{q}} + s_n(\mathbf{q}, \dot{\mathbf{q}}) \geq 0 \tag{3.26}$$

$$R_t(\mathbf{q})\ddot{\mathbf{q}} + s_t(\mathbf{q}, \dot{\mathbf{q}}) = 0 \tag{3.27}$$

$$\lambda_n \geq 0 \tag{3.28}$$

$$\lambda_n^T[R_n(\mathbf{q})\ddot{\mathbf{q}} + s_n(\mathbf{q}, \dot{\mathbf{q}})] = 0 \tag{3.29}$$

where $R_n(q)\lambda_n + R_t(q)\lambda_t$ represents the contact forces/torques acting on the robot. Their magnitude is captured by λ_n and λ_t (the Lagrange multipliers).

3.1.2.4 Single Support Phase (SSP)

Consider the robot standing on one foot while the other is swinging. Let $\mathcal{F}_{st}$ be the reference frame attached the the stance foot (see Figure 3.2) and let us assume that a minimum of three non-collinear points of its flat foot are in contact with the ground. The reaction forces acting on these points will yield a reaction wrench which could be expressed in $\mathcal{F}_{st}$ as [111]

$$\mathcal{T}_{st} \triangleq \begin{bmatrix} f_{st}^x & f_{st}^y & f_{st}^z & m_{st}^x & m_{st}^y & m_{st}^z \end{bmatrix}^T \tag{3.30}$$

where the f_{st}^i and m_{st}^i denote the net force and net moment respectively along and about the i direction.

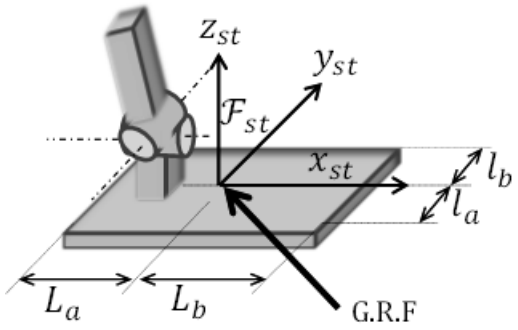


Figure 3.2: Stance foot with its geometrical parameters, the assigned frame and the ground reaction forces (G.R.F)

The dynamic model under this reaction wrench will be given by [124]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{\Gamma} + \mathbf{J}_{st}^T(\mathbf{q})\mathcal{T}_{st} \tag{3.31}$$

where $\mathbf{J}_{st}^T(\mathbf{q})$ is the Jacobian of $\mathcal{F}_{st}$ (where $\mathcal{T}_{st}$ is applied).

Solving the dynamics (3.31) requires additional equation, since $\mathcal{T}_{st}$ is unknown (unless measured). This could be obtained from the constraints imposed by the contacts. Let us define

$$\boldsymbol{P}_{st} \triangleq \begin{bmatrix} t_{st}^T & \phi_{st}^T \end{bmatrix}^T \tag{3.32}$$

as the pose of the stance foot with respect to $\mathcal{F}_W$. If in addition to the contacts assumption, it is assumed no slipping and no rotation of the stance foot, then the pose $\boldsymbol{P}_{st}$ is constrained to remain constant. Therefore, this results in constraints also on velocity and acceleration such that

$$\mathbf{J}_{st}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}_{6\times 1} \tag{3.33}$$

$$\mathbf{J}_{st}(\mathbf{q})\ddot{\mathbf{q}} + \frac{d}{dt}\left(\mathbf{J}_{st}(\mathbf{q})\right)\dot{\mathbf{q}} = \mathbf{0}_{6\times 1} \tag{3.34}$$

Now using Equations (3.31) and (3.34), the joint acceleration $\ddot{\mathbf{q}}$ and the wrench $\mathcal{T}_{st}$ can be computed as function of the state $(\mathbf{q}, \dot{\mathbf{q}})$ and the applied torque $\mathbf{\Gamma}$.

Walking Constraints Owing to its unilateral nature, the normal component of $\mathcal{T}_{st}$ (f_{st}^z) has to satisfy the constraint

$$f_{st}^z > 0 \tag{3.35}$$

while the no-slipping condition, according to (3.24), imposes the following constraint on the tangential forces

$$\sqrt{(f_{st}^x)^2 + (f_{st}^y)^2} < \mu_f \, f_{st}^z \tag{3.36}$$

To linearize the constraint (3.36), the friction cone can be approximated by a convex polyhedron [133], for instance a friction pyramid [111], which gives

$$|f_{st}^x| < \frac{\mu_f}{\sqrt{2}} f_{st}^z, \quad \text{and} \quad |f_{st}^y| < \frac{\mu_f}{\sqrt{2}} f_{st}^z \tag{3.37}$$

The ground reaction moments were also shown to be bounded because of the finite size of the feet in conjunction with unilateral contact forces. This could be written as [111]

$$\begin{cases} -l_b f_{st}^z < m_{st}^x < l_a f_{st}^z \\ -L_a f_{st}^z < m_{st}^y < L_b f_{st}^z \end{cases} \tag{3.38}$$

where L_a, L_b, l_a and l_b geometrical quantities of the foot as shown in Figure 3.2.

The constraints (3.35), (3.37), and (3.38) can be gathered as follows

$$\mathcal{A}_{\mathcal{F}_{st}}^{T}(\mathbf{q})\mathcal{T}_{st} > \mathbf{0} \tag{3.39}$$

Finally, the dynamic model of the humanoid robot in single support phase while considering the walking constraints will be given by

$$\begin{cases} \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{\Gamma} + \mathbf{J}_{st}^{T}(\mathbf{q})\mathcal{T}_{st} \\ \mathbf{J}_{st}(\mathbf{q})\ddot{\mathbf{q}} + \frac{d}{dt}\left(\mathbf{J}_{st}(\mathbf{q})\right)\dot{\mathbf{q}} = \mathbf{0}_{6\times 1} \\ \mathcal{A}_{\mathcal{F}_{st}}^{T}(\mathbf{q})\mathcal{F}_{st} > \mathbf{0} \end{cases} \tag{3.40}$$

This SSP model will stay valid as long as $\mathcal{T}_{st}$ satisfies the constraints (3.39).

3.1.2.5 Impact Phase Model

During walking the impact phase occurs when a foot strikes the ground with a nonzero velocity, for instance while switching the support foot. If this transition is not instantaneous and both feet are simultaneously in contact with the ground, then the robot becomes in double support phase (DSP) whose dynamic model is given by

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{\Gamma} + \mathbf{J}_{lf}^{T}(q)\mathcal{T}_{lf} + \mathbf{J}_{rf}^{T}(q)\mathcal{T}_{rf} \tag{3.41}$$

where $\mathbf{J}_{lf}(q)$ and $\mathbf{J}_{rf}(q)$ are respectively the Jacobian of left and right foot, while $\mathcal{T}_{lf}$ and $\mathcal{T}_{rf}$ denote respectively the reaction wrench acting on the left and right foot. The solution of this dynamics requires the constraints (holonomic, kinematic, unilateral, etc.) associated with each foot to be considered, and they can be different from one foot to the other [111].

In our case, we will assume an instantaneous DSP, which means that after the impact phase the support foot instantaneously becomes the swing foot. The impact is assumed to be inelastic and non-sliding. The contact wrench acting over an infinitesimal interval of time yields a jump of the system's velocities while the configuration is assumed to stay unchanged. Provided that the actuators do not generate impulse forces/torques, it can be shown that the impact dynamics will be given by [131]

$$\mathbf{M}(\mathbf{q})\left(\dot{\mathbf{q}}^{+} - \dot{\mathbf{q}}^{-}\right) = \mathbf{J}_{st}^{T}(\mathbf{q})\mathcal{T}_{imp_{st}} \tag{3.42}$$

with $(\mathbf{q}^{+} = \mathbf{q}^{-} = \mathbf{q})$, and where $(\mathbf{q}^{-}, \dot{\mathbf{q}}^{-})$ and $(\mathbf{q}^{+}, \dot{\mathbf{q}}^{+})$ denote respectively the states before and after the impact, $\mathcal{T}_{imp_{st}} \triangleq \int_{t^{-}}^{t^{+}} \delta f_{imp_{st}}(\tau)\, d\tau$ is the magnitude of the impulsive contact wrench over the small time interval $[t^{-}, t^{+}]$. $\delta f_{imp_{st}}$ is the model of the impulsive wrench, with δ denoting the Dirac impulse function.

Under the assumption that the impact occurs on the swing foot's sole, which remains static (no slipping, no rotation), the following constraint holds

$$\mathbf{J}_{st}(q)\dot{\mathbf{q}}^{+} = \mathbf{0} \tag{3.43}$$

where $\mathbf{J}_{st}(q)$ is the Jacobian of the new stance foot (former swing foot).

Finally, the impulsive contact wrench $\mathcal{T}_{imp_{st}}$ and the velocity after impact $\dot{\mathbf{q}}^{+}$ are computed from Equation (3.42) in conjunction with (3.43). It can be shown that the resulting solution is [134, 111]

$$\begin{aligned} \mathcal{T}_{imp_{st}} &= -\left(\mathbf{J}_{st}\mathbf{M}(\mathbf{q})^{-1}\mathbf{J}_{st}^{T}\right)\mathbf{J}_{st}\dot{\mathbf{q}}^{-} \\ \dot{\mathbf{q}}^{+} &= \left[\mathbf{I}_{N+6} - \mathbf{M}(\mathbf{q})^{-1}\mathbf{J}_{st}^{T}\left(\mathbf{J}_{st}\mathbf{M}(\mathbf{q})^{-1}\mathbf{J}_{st}^{T}\right)\mathbf{J}_{st}\right]\dot{\mathbf{q}}^{-} \end{aligned} \tag{3.44}$$

Remark The Jacobian $\mathbf{J}_{st}(\mathbf{q})$ is computed similarly to $\mathbf{J}_{i}(\mathbf{q})$ given by (3.13) and (3.14); it is obtained as follows

$$\mathbf{J}_{st}(\mathbf{q}) = \begin{bmatrix} \begin{bmatrix} \mathbf{R}(\phi_b) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{R}(\phi_b) \end{bmatrix}\mathbf{J}_{st}(\mathbf{q}_{hr}) & \begin{matrix} \mathbf{I}_{3\times3} & -\left[\mathbf{R}(\phi_b)\boldsymbol{r}_{st}\right]_{\times}\mathbf{T}(\phi_b) \\ \mathbf{0}_{3\times3} & \mathbf{T}(\phi_b) \end{matrix} \end{bmatrix} \tag{3.45}$$

where $\mathbf{J}_{st}(\mathbf{q}_{hr})$ and $\boldsymbol{r}_{st}$ represent respectively the Jacobian and the position vector of $\mathcal{F}_{st}$ expressed in the robot base frame $\mathcal{F}_b$.

3.2 Balance and Locomotion Control

The posture of a walking robot is said to be balanced and its gait stable if the robot can stand and walk without falling [135]. The role of the control scheme is then to compute and coordinate at every instant, depending on the current state of the robot, motions able to maintain postural balance or stability of the gait.

However, it was shown [16] that a desired trajectory can only be realized by a biped robot if and only if the dynamic wrench of the latter is equal to the total wrench due to gravity and contact forces. In addition, the contact forces must satisfy a number of constraints [133]. It is therefore the duty of the control scheme to ensure that this condition is satisfied.

In the particular case where the accelerations and velocities are small, the dynamic wrench becomes negligible with respect to the gravity and contact wrenches. Then, the condition above is reduced to require that the projection of the robot's center of mass (CoM) lies in convex hull

of contact points. This is defined as *static stability*. Thus, we can infer that a stable walking achieved with a non-negligible dynamic wrench is *dynamically stable*. Different concepts have been proposed to ensure such a stability; the most commonly used is the Zero-Moment-Point (ZMP) [17].

3.2.1 ZMP based Dynamic Stability

Given the fact that the fall of a robot supposes the existence of tipping moment(s) [112], the concept of bipedal stability involves directly the notion of moments. In case all contact points lie on a planar horizontal ground, *Vukobratovic* and *Stepanenko* [17] showed that there exist a point on the ground where the moments around the horizontal axes (tipping moments) are zero; this point was called the Zero-Moment-Point (ZMP), it is also the Center of Pressure (CoP). The ZMP must lie within the convex hull of contact points for walking to be stable. It can be estimated using measurements or computed using the humanoid's dynamics.

3.2.1.1 Measurement based ZMP Computation

The ZMP represents the application point of the resultant of unilateral reaction forces acting on the foot [17]. Assuming N contact points between the foot and the ground, the ZMP can be obtained as follows

$$\boldsymbol{p} \triangleq \frac{\sum_{i=1}^{N} \boldsymbol{p}_i f_{iz}}{\sum_{i=1}^{N} f_{iz}}, \tag{3.46}$$

where $\boldsymbol{p}_i \triangleq [\ p_{ix} \quad p_{iy} \quad p_{iz}\ ]^T$ represents the position vector of the contact point i, f_{iz} is the z (vertical) component of the contact force i, defined as $\boldsymbol{f}_i \triangleq [\ f_{ix} \quad f_{iy} \quad f_{iz}\ ]^T$. Due to the unilateral nature of the ground reaction forces, it can be shown that the ZMP can only exist within the the support polygon [135].

The moment of forces about the ZMP can be computed as follows

$$\boldsymbol{\tau}_{ZMP} = \sum_{i=1}^{N} (\boldsymbol{p}_i - \boldsymbol{p}) \times \boldsymbol{f}_i, \tag{3.47}$$

where the moment is defined as $\boldsymbol{\tau}_{ZMP} \triangleq [\ \tau_{ZMP_x} \quad \tau_{ZMP_y} \quad \tau_{ZMP_z}\ ]^T$. Developing Equation (3.47) in conjunction with (3.46), results in

$$\tau_{ZMP_x} = \tau_{ZMP_y} = 0 \tag{3.48}$$

for a horizontal ground ($p_{iz} = p_z$), hence the *zero-moment point* name.

Though the moment about the vertical axis, τ_{ZMP_z}, might be different from zero, it cannot tip over the robot. τ_{ZMP_z} can merely change the robot's direction.

3.2.1.2 Dynamics based ZMP Computation

Unlike the method (3.46), which uses the measurements of the reaction forces in order to determine the ZMP, the robot dynamics can be used to compute or even predict the resulting ZMP. This possibility is more interesting from the control point of view, since it allows one to design motions in order to obtain a desired ZMP and thus, to guarantee dynamic walking.

With reference to Figure 3.3, let $\mathcal{P} \triangleq [\ \mathcal{P}_x \quad \mathcal{P}_y \quad \mathcal{P}_z\ ]^T$ and $\mathcal{L} \triangleq [\ \mathcal{L}_x \quad \mathcal{L}_y \quad \mathcal{L}_z\ ]^T$ denote respectively the linear and angular momenta, which are given by

$$\mathcal{P} = \sum_{i=1}^{N_l} m_i \dot{\boldsymbol{c}}_i, \tag{3.49}$$

$$\mathcal{L} = \sum_{i=1}^{N_l} \left[\boldsymbol{c}_i \times (m_i \dot{\boldsymbol{c}}_i) + \mathbf{R}_i \mathbf{I}n_i \mathbf{R}_i^T \boldsymbol{\omega}_i \right], \tag{3.50}$$

where m_i, $\boldsymbol{c}_i$, and $\boldsymbol{\omega}_i$ are respectively the mass, the position of the CoM, and the angular velocity of the i^{th} link, all expressed with respect to a fixed inertial frame. $\mathbf{I}n_i$ denotes the i^{th} link's inertia tensor expressed in the link frame, and $\mathbf{R}_i$ is the rotation matrix from the i^{th} link's frame to the fixed inertial frame, such that $\mathbf{R}_i \mathbf{I}n_i \mathbf{R}_i^T$ expresses the inertia tensor in the fixed frame.

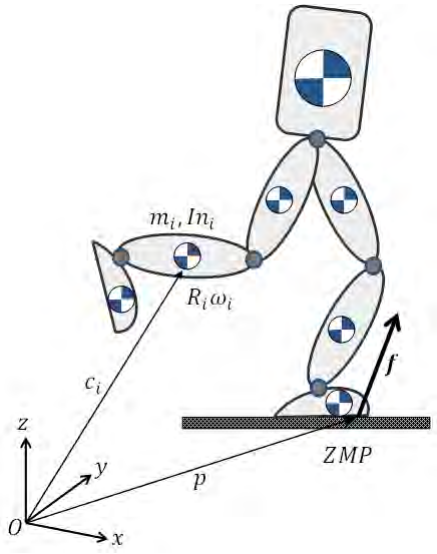


Figure 3.3: Dynamic model of humanoid robot for ZMP computation

If the external force is assumed to act on the ZMP $\boldsymbol{p}$ (see Figure 3.3), it can be shown [135] that the moment about the ZMP will be given by

$$\tau_{ZMP} = \underbrace{\dot{\mathcal{L}} - \boldsymbol{c} \times M\boldsymbol{g}}_{\boldsymbol{\tau}} + \underbrace{(\dot{\mathcal{P}} - M\boldsymbol{g})}_{\boldsymbol{f}} \times \boldsymbol{p}, \tag{3.51}$$

where $\boldsymbol{f}$ and $\boldsymbol{\tau}$, according to Newton-Euler's law on the variation of the linear and angular

momenta, represent the external force and torque applied on the system, respectively. $\boldsymbol{g} = [\ 0 \quad 0 \quad -g\ ]^T$ is the gravity acceleration vector, while $M = \sum_{i=1}^{N_l} m_i$ and $\boldsymbol{c} = \sum_{i=1}^{N_l} \frac{m_i \boldsymbol{c}_i}{M}$ are the total mass and the position vector of the CoM of the robot, respectively.

After developing Equation (3.51), and equating τ_{ZMP_x} and τ_{ZMP_y} to zero, the ZMP is obtained as

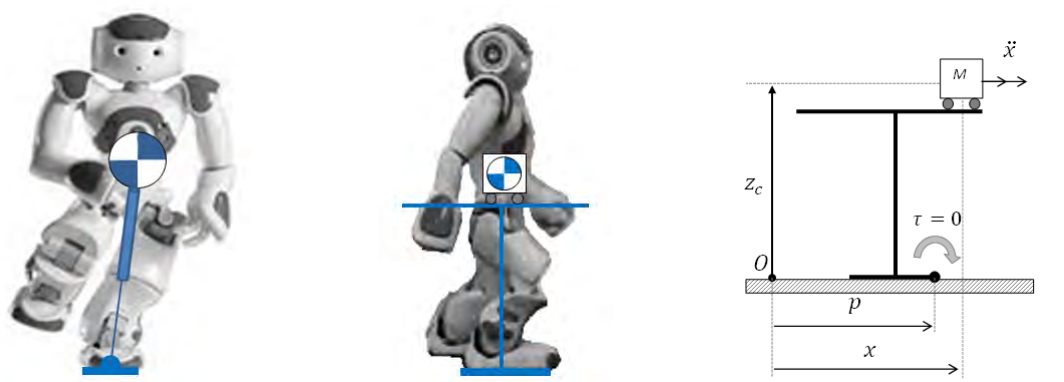
$$p_x = \frac{Mgc_x + p_z\dot{\mathcal{P}}_x - \dot{\mathcal{L}}_y}{Mg + \dot{\mathcal{P}}_z} \tag{3.52}$$

$$p_y = \frac{Mgc_y + p_z\dot{\mathcal{P}}_y + \dot{\mathcal{L}}_y}{Mg + \dot{\mathcal{P}}_z} \tag{3.53}$$

where c_i and p_i are the components of $\boldsymbol{c} \triangleq [\ c_x \quad c_y \quad c_z\ ]^T$ and $\boldsymbol{p} \triangleq [\ p_x \quad p_y \quad p_z\ ]^T$, respectively.

3.2.2 Walking Control using Simplified Model

To control a humanoid robot, its motions should be generated in real-time in accordance with its dynamics. Although more accurate, using the real humanoid dynamics requires a high computational cost. To lower this cost, common approaches rely on approximate models, especially when the robot stands on one leg. The idea is to use mass concentrated models, such as the Inverted Pendulum Model with point mass [136, 137, 138] or with flywheel [139], or the Cart-Table model [113].



(a) Inverted pendulum model

(b) Cart-Table model

(c) ZMP with Cart-Table model [Kajita 2003]

Figure 3.4: Linear mass concentrated models

3.2.2.1 Linear Inverted Pendulum Model

The 3D Linear Inverted Pendulum Model (3D-LIPM) was proposed by *Kajita et al.*, [137] based on the fact that a humanoid robot standing on one leg can be modeled as a three dimensional inverted pendulum, with the robot's CoM connected to a massless leg (see Figure 3.4a). The

obtained linear dynamics stems from the idea to constrain the motion onto an arbitrary plane. When the constraint plane is horizontal, it was shown [140] that the LIPM equations become

$$\ddot{c}_x = \frac{g}{z_c} c_x + \frac{1}{M z_c} \tau_y \tag{3.54}$$

$$\ddot{c}_y = \frac{g}{z_c} c_y - \frac{1}{M z_c} \tau_x \tag{3.55}$$

with $z_c = c_z$ the height of the CoM defined by the constraint plane. τ_x and τ_y are respectively the torques around the x and y axis. The ZMP definition applied to (3.54) and (3.55) gives [113, 138]

$$\ddot{c}_x = \frac{g}{z_c}(c_x - p_x) \tag{3.56}$$

$$\ddot{c}_y = \frac{g}{z_c}(c_y - p_y) \tag{3.57}$$

3.2.2.2 Cart-Table Model

The cart-table model [113] is dual to the 3D LIPM; it simplifies a walking robot's dynamics by modeling its CoM as a running cart on a massless pedestal table (see Figure3.4b). Within this representation, the ZMP appears clearly. The moment about it is given by

$$\tau_y = -Mg(c_x - p_x) + M\ddot{c}_x z_c. \tag{3.58}$$

A similar equation can be written for the lateral plane of the robot, thus yielding two equations. The ZMP is obtained for $\tau_y = \tau_x = 0$, hence

$$p_x = c_x - \frac{z_c}{g}\ddot{c}_x \tag{3.59}$$

$$p_y = c_y - \frac{z_c}{g}\ddot{c}_y \tag{3.60}$$

Note that if the computed ZMP lies inside the support polygon, the walking motion is guaranteed to satisfy the unilateral constraint with full foot contact with the ground [135]. Equations (3.59) and (3.60), however, do not explicitly account for the support polygon, since one can arbitrarily find values of c_i and $\ddot{c}_i$ $(i = x, y)$ giving a ZMP outside the support polygon (for instance, the one in Figure 3.4c). In such a case, the walking motion will be compromised.

3.3 Reactive Omnidirectional Walking Pattern Generation

The locomotion control problem is usually solved by generating trajectories specifying how the biped robot should move for stable walking. Given a walking trajectory specified, for instance, by a time sequence of the footsteps within which the ZMP lies at the center. Using the simplified model (3.59)-(3.60), the pattern generation problem is to determine the CoM's motion that will satisfy the desired ZMP.

In view of these equations, *Kajita et al.*, [113] proposed to solve this problem as a servo control problem, with the CoM's jerk defined as input. Thus, the state variables of the obtained dynamical system are nothing but the CoM's motion yielding the desired ZMP.

However, a valid motion requires that the CoM starts moving before the desired ZMP; in other words, the CoM should start changing its state with respect to the future input (ZMP reference). Hence, *Kajita et al.*,[113] proposed a solution with a *preview controller* [141]. *Wieber et al.*, [142] instead proposed a Linear Model Predictive Controller (LMPC) offering the ability to handle constraints. Extending the original MPC based formulation, *Diedam et al.* [143] proposed a *continuous adaptation of footsteps position*. Based on the latter formulation, *Herdt et al.* [23, 27] developed a pattern generator allowing *automatic footsteps placement* while following reference translational velocities, in [27] an extension to rotational velocity is proposed. However, the extension proposed in the latter reference introduced non-linearities in the pattern generation to the extent that the authors rather preferred a predetermination of rotation [23].

In this section, we propose a new extension for rotational motions on the automatic footsteps placement pattern generator. The proposed approach removes completely the need of any predetermination and therefore makes the pattern generator more reactive.

3.3.1 Predictive Control Based Pattern Generation

Following the servo control approach [113], Equations (3.59)-(3.60) can be written in discrete state space form as

$$\begin{cases} \widehat{c}^h_{k+1} & = A\,\widehat{c}^h_k + B\dddot{c}^h_k \\ p^h_k & = C\,\widehat{c}^h_k \end{cases}, \tag{3.61}$$

where the states vector $\hat{c}^h_k$ is

$$\hat{c}^h_k = \begin{bmatrix} c^h(kT) \\ \dot{c}^h(kT) \\ \ddot{c}^h(kT) \end{bmatrix} \quad , \quad \text{with } h = x, y \tag{3.62}$$

and the matrix A, and the vectors B and C are

$$A = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{T^3}{6} \\ \frac{T^2}{2} \\ T \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & -\frac{z_c}{g} \end{bmatrix} \tag{3.63}$$

with T denoting the sampling time.

An LMPC formulation of the CoM and the ZMP dynamics (3.61) over a prediction horizon N_p can be written as [142]

$$\widehat{C}^h_{k+1} = S_c\,\hat{c}^h_k + U_c\,\dddot{C}^h_k \tag{3.64}$$

for the CoM dynamics, and

$$P_{k+1}^{h} = S_{p_h}\hat{c}_k^h + U_{p_h}\dddot{C}_k^h \tag{3.65}$$

for the ZMP dynamics, where $\hat{c}_k^h = \begin{bmatrix} c^h(k) & \dot{c}^h(k) & \ddot{c}^h(k) \end{bmatrix}^T$ is the states vector, with $h \equiv x,\, y$. $\widehat{C}_{k+1}^h \in \mathbb{R}^{3N_p \times 1}$ and $P_{k+1}^h \in \mathbb{R}^{N_p \times 1}$ denote, respectively, the predicted states vector (CoM motion) and the output vector (ZMP position) over the horizon N_p defined as

$$\widehat{C}_{k+1}^h = \begin{bmatrix} c^h(k+1 \mid k) \\ \vdots \\ c^h(k+N_p \mid k) \end{bmatrix}, \quad P_{k+1}^h = \begin{bmatrix} p^h(k+1 \mid k) \\ \vdots \\ p^h(k+N_p \mid k) \end{bmatrix}, \tag{3.66}$$

and $\dddot{C}_k^h \in \mathbb{R}^{N_c \times 1}$ is the input (jerk) vector, with N_c denoting the control horizon. Note that $S_c \in \mathbb{R}^{3N_p \times 3}$, $U_c \in \mathbb{R}^{3N_p \times N_c}$, $S_{p_h} \in \mathbb{R}^{N_p \times 3}$, and $U_{p_h} \in \mathbb{R}^{N_p \times N_c}$ are matrices resulting from the LMPC formulation [142].

At this stage, if an objective function is designed to minimize the jerk, $\left\|\dddot{C}_k^h\right\|$, and the error between the ZMP and its reference over the horizon N_p, $\left\|P_{k+1}^h - P_{k+1}^{h\_ref}\right\|$, this scheme will result in the LMPC based pattern generator developed by *Wieber et al.* in [142], where the ZMP's reference positions are still specified in advance (by the steps planner). Though *Wieber*'s pattern generator performs similarly to the preview controller based pattern generator [113], their scheme has the ability to handle different walking constraints.

A typical locomotion control scheme using this approach is illustrated in Figure 3.5, where it is usually implemented with a feedback stabilization to improve robustness to model uncertainty. This feedback stabilization can be realized directly in the pattern generator by using the true states instead of the model states. In this case, it may require a states observer [112].

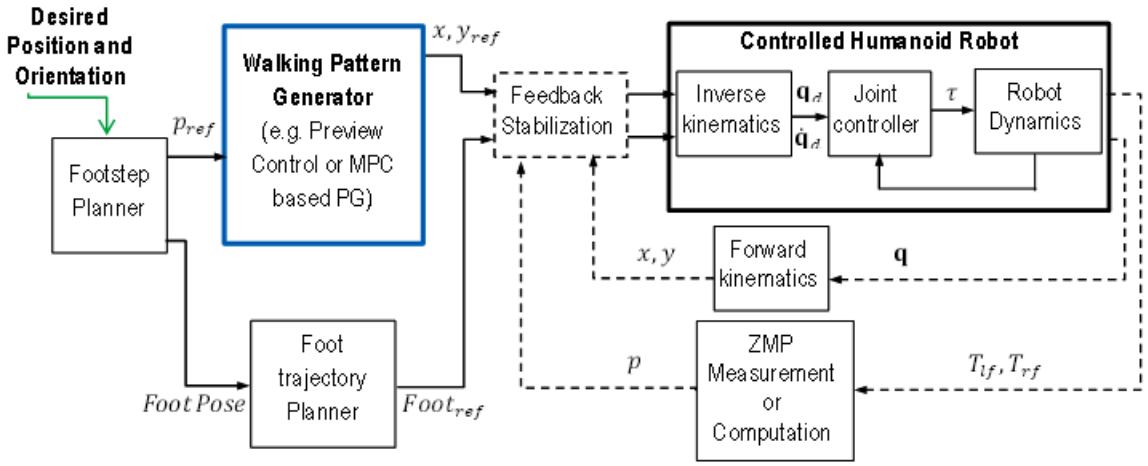


Figure 3.5: Locomotion control of humanoid robot with footstep planner

As stated in the introductory chapter, because it requires knowledge beforehand of the footsteps, this pattern generator is convenient for positioning tasks. However, its performances will decrease for reactive tasks, for example those performed with visual servoing. It therefore becomes important to allow self-adaptation of the footsteps.

3.3.2 Automatic Footsteps Placement

In order to generate automatically the footstep placements, it was proposed in [143, 23, 27] to let the optimization process decide the footstep positions. This was achieved by defining new variables associated with the footsteps taking place over the prediction horizon and by requiring the CoM to follow a desired velocity.

Thus, let us define ${}^{w}f_{k}^{h}$ and ${}^{w}f_{k+1}^{h}$, respectively as the current pose of the stance foot, and the poses of the m following steps on the ground with respect to a fixed inertial frame $\mathcal{F}_W$. It was shown that [23] the overall feet poses, ${}^{w}F_{k+1}^{h}$, over the prediction horizon can be written in compact form as

$$ {}^{w}\boldsymbol{F}_{k+1}^{h} = V_{k+1}^{c}\,{}^{w}f_{k}^{h} + V_{k+1}^{f}\,{}^{w}\boldsymbol{f}_{k+1}^{h} \tag{3.67} $$

where h represents the pose parameters x and y when assuming an horizontal ground. V_{k+1}^{c} and V_{k+1}^{f} are selection matrices (whose elements are ones and zeros) associating steps to corresponding sampling time. Let us also rewrite (3.64) in three separated equations, describing the position, the velocity, and the acceleration of the CoM over the prediction horizon, respectively. That is

$$ C_{k+1}^{h} = S_h \hat{c}_k^h + U_h \dddot{C}_k^h \tag{3.68} $$

$$ \dot{C}_{k+1}^{h} = S_{\dot{h}} \hat{c}_k^h + U_{\dot{h}} \dddot{C}_k^h \tag{3.69} $$

$$ \ddot{C}_{k+1}^{h} = S_{\ddot{h}} \hat{c}_k^h + U_{\ddot{h}} \dddot{C}_k^h \tag{3.70} $$

where S_h, $S_{\dot{h}}$, and $S_{\ddot{h}} \in \mathbb{R}^{N_p \times 3}$ derive from $S_c \in \mathbb{R}^{3N_p \times 3}$ and U_h, $U_{\dot{h}}$, and $U_{\ddot{h}} \in \mathbb{R}^{N_p \times N_c}$ derive from $U_c \in \mathbb{R}^{3N_p \times N_c}$, (see ref. [143], [23] for more details).

Consider now a robot's walking objective, which consists of following a reference velocity continuously or at least on average, with the feet positions centered around the ZMP (CoP). If additionally, smooth robot's trajectories are desired, *Herdt et al.*, [23] showed that the walking motion with automatic footsteps placement can result from the following optimization problem

$$ \begin{aligned} \min_{u_k} \sum_{h=x,y} \Bigg\{ & \frac{\sigma}{2} \left\| \dot{C}_{k+1}^{h} - \dot{C}_{ref}^{h} \right\|^2 + \frac{\varepsilon}{2} \left\| E C_{k+1}^{h} - \dot{C}_{ref}^{h} \right\|^2 \\ & + \frac{\kappa}{2} \left\| P_{k+1}^{h} - {}^{w}F_{k+1}^{h} \right\|^2 + \frac{\mu}{2} \left\| \dddot{C}_{k+1}^{h} \right\|^2 \Bigg\} \end{aligned} \tag{3.71} $$

where μ, σ, κ and ε are weighting factors, $\dot{C}_{ref}^{h}$ represents the reference velocity of the CoM, E is a double diagonal matrix used to compute the average velocity of the CoM from its positions

over two steps, and the vector u_k is defined as

$$u_k \triangleq \begin{bmatrix} \dddot{C}^x_{k+1} \\ {}^w f^x_{k+1} \\ \dddot{C}^y_{k+1} \\ {}^w f^y_{k+1} \end{bmatrix}. \tag{3.72}$$

From (3.71) subject to stability constraints (the ZMP must lie within the support polygon), constraints to prevent self-collision, legs overstretching, or to limit the velocity, etc. (we recommend ref. [143], [23] for more details), the resulting optimal solutions of ${}^w f^x_{k+1}$ and ${}^w f^y_{k+1}$ are nothing but the coordinates of the desired footstep placements satisfying the walking objective.

Thus, at every new step, ${}^w f^h_k$ in (3.67) is updated with the optimal ${}^w f^h_{k+1}$, allowing as such the cycle to continue. As the result, the locomotion control does no longer need a footstep planner; providing reference velocity yields an autonomous walk as illustrated in Figure 3.6.

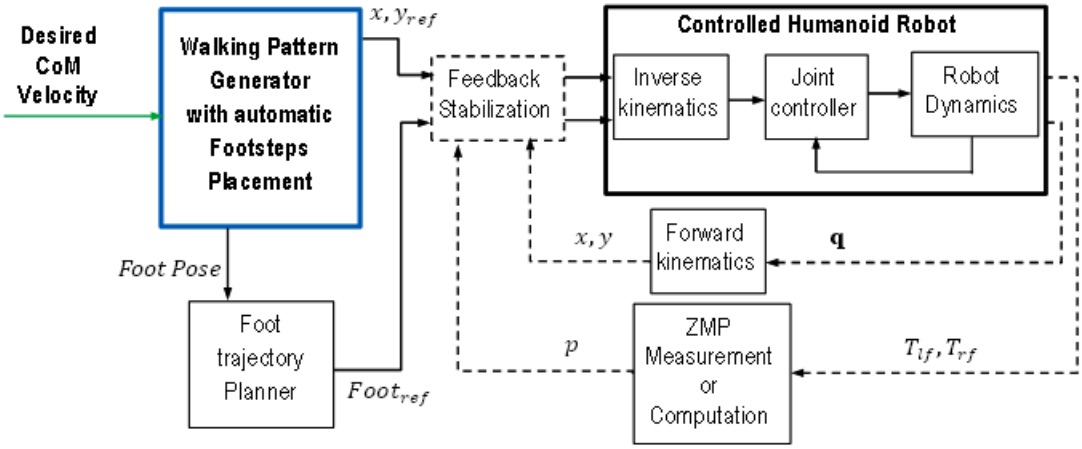


Figure 3.6: Locomotion Control with automatic footsteps placement

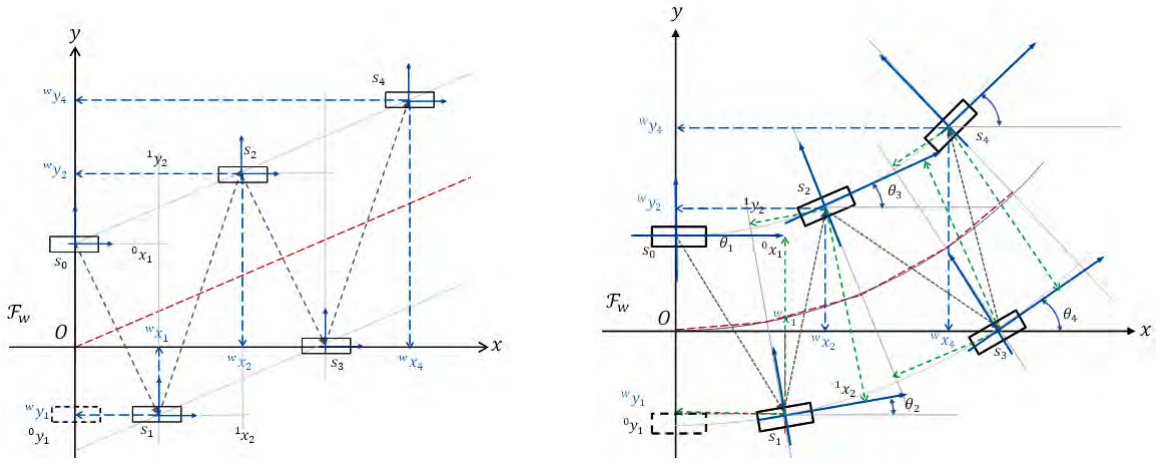
Note that the optimization (3.71) applies to translational motions. In order to follow rotational velocities, *Herdt et al.*, [27] suggested, as an extension to (3.71), the additional optimization problem

$$\min_{\dddot{C}^\theta_k} \frac{\sigma}{2} \left\| \dot{C}^\theta_{k+1} - \dot{C}^\theta_{ref} \right\|^2 + \frac{\varepsilon}{2} \left\| EC^\theta_{k+1} - \dot{C}^\theta_{ref} \right\|^2 + \frac{\kappa}{2} \left\| \sum_i \left({}^w f^\theta_i - c^\theta_i \right) \right\|^2 \tag{3.73}$$

where C^θ_{k+1} and its derivatives, similarly derived as (3.68)-(3.70), describe the rotational motion of the CoM over N_p. $\dot{C}^\theta_{ref}$ represents the reference rotational CoM's velocity, while ${}^w f^\theta_i$ and c^θ_i denote the orientations respectively of the support foot and of the frame attached to the CoM, with the subscript i denoting a previewed time instant. However, they did not use (3.73) for their results, arguing that it introduces nonlinear constraints. They therefore chose to "*predefine the orientations of the feet and the trunk*" before solving (3.71) [27].

3.3.3 Automatic Footsteps Placement and Orientation

In this subsection, we propose an extension to *Herdt*'s pattern generator, previously described, with the aim to enhance its reactive performances particularly for rotational velocities. That is: overcoming the predetermination of rotation, while avoiding or minimizing the subsequent non-linearity introduced. To that end, let us consider Figure 3.7 depicting two walking trajectories, a pure translation in Figure 3.7a and combined translation and rotation in Figure 3.7b.



(a) Pure translation

(b) Translation and rotation

Figure 3.7: Footsteps and walking trajectories

If the positions of the future footsteps have to be described with respect to $\mathcal{F}_W$ while accounting for the support foot orientation, Equation (3.67) has to be reformulated, since the translations along x and y will be geometrically coupled due to the rotation. We have, for instance

$$\begin{aligned} {}^{w}x_1 &= {}^{w}x_0 + {}^{0}x_1 cos\theta_1 - {}^{0}y_1 sin\theta_1 \\ {}^{w}y_1 &= {}^{w}y_0 + {}^{0}x_1 sin\theta_1 + {}^{0}y_1 cos\theta_1 \end{aligned} \tag{3.74}$$

where $({}^{w}x_1, {}^{w}y_1)$ and $({}^{w}x_0, {}^{w}y_0)$ with reference to Figure 3.7b, represent the positions of the footsteps s_1 respectively s_0 with respect to $\mathcal{F}_W$. $({}^{0}x_1, {}^{0}y_1)$ denotes the position of s_1 relative to the frame attached to s_0, and θ_1 represents the orientation of the frame s_0 relative to $\mathcal{F}_W$.

Given the fact that in absence of slippage, the orientation of the stance foot with respect to a fixed inertial frame, for instance $\mathcal{F}_W$, is constant during a step period, similarly to the positions, the steps orientations over the prediction horizon can be modeled by (3.67) as follows

$${}^{w}\boldsymbol{\Theta}^{ref}_{k+1} = V^{c}_{k+1}\,{}^{w}\theta^{c}_{k} + V^{f}_{k+1}\,{}^{w}\boldsymbol{\theta}^{f}_{k+1} \tag{3.75}$$

where ${}^{w}\theta^{c}_{k}$ and ${}^{w}\boldsymbol{\theta}^{f}_{k+1}$ denote respectively, the orientation of the current stance foot and the vector of orientations of the m future steps.

Following Equation (3.74), let us consider now two complete walking cycles ($m = 3$), it can be shown that the positions of the four steps with respect to $\mathcal{F}_W$ can be written as

$$\mathbf{F}^w_{s_i+1} = \mathbf{I}_f.F^w_{s_i} + \mathbf{R}_{xy}\triangle\mathbf{F}_{s_i} \tag{3.76}$$

where $F^w_{s_i} \triangleq \begin{bmatrix} {}^w f^x_k & {}^w f^y_k \end{bmatrix}^T$ is the position of step s_i with respect to $\mathcal{F}_w$, $\mathbf{I}_f \in \mathbb{R}^{8\times 2}$ is a superposition of four 2×2 identity matrices, and the other variables are defined as follows

$$\mathbf{F}^w_{s_i+1} \triangleq \begin{bmatrix} F^w_{s_i} \\ F^w_{s_{i+1}} \\ F^w_{s_{i+2}} \\ F^w_{s_{i+3}} \end{bmatrix}, \mathbf{R}_{xy} \triangleq \begin{bmatrix} 0 & 0 & 0 \\ R^w_{s_i} & 0 & 0 \\ R^w_{s_i} & R^w_{s_{i+1}} & 0 \\ R^w_{s_i} & R^w_{s_{i+1}} & R^w_{s_{i+2}} \end{bmatrix}, \tag{3.77}$$

$$\triangle\mathbf{F}_{s_i} \triangleq \begin{bmatrix} F^{s_i}_{s_{i+1}} \\ F^{s_{i+1}}_{s_{i+2}} \\ F^{s_{i+2}}_{s_{i+3}} \end{bmatrix}, \; R^w_{s_i} \triangleq \begin{bmatrix} cos\theta_{s_i} & -sin\theta_{s_i} \\ sin\theta_{s_i} & cos\theta_{s_i} \end{bmatrix} \tag{3.78}$$

where $\triangle\mathbf{F}_{s_i} \in \mathbb{R}^6$ is the vector of relative footsteps positions, such that

$$F^{s_i}_{s_{i+1}} = \begin{bmatrix} f^{x\_s_i}_{s_{i+1}} & f^{y\_s_i}_{s_{i+1}} \end{bmatrix}^T$$

denotes the position of the step s_{i+1} relative to s_i, $R^w_{s_i}$ and θ_{s_i} are respectively the orientation matrix and orientation angle of the foot at step s_i with respect to $\mathcal{F}_w$. Note that with this formulation, one can recursively determine the positions of n steps with respect to the fixed inertial frame $\mathcal{F}_w$, knowing only the relative steps positions and orientations, which can easily be obtained from the forward kinematics.

Assume now that s_i represents the current stance foot, whose position is given by $\begin{bmatrix} {}^w f^x_k & {}^w f^y_k \end{bmatrix}^T$, then the steps s_{i+1}, s_{i+2} and s_{i+3} will represent the previewed (future) steps, whose positions can be denoted by the vector ${}^w f^h_{k+1} \in \mathbb{R}^6$. With these considerations, using Equations (3.76)-(3.78), the x and y components of the three future steps vector ${}^w f^h_{k+1}$ can be written separately as follows

$$\begin{cases} {}^w f^x_{k+1} &= \mathbf{1}_3.{}^w f^x_k + \mathbf{R}_x\triangle\mathbf{F}_{s_i} \\ {}^w f^y_{k+1} &= \mathbf{1}_3.{}^w f^y_k + \mathbf{R}_y\triangle\mathbf{F}_{s_i} \end{cases} \tag{3.79}$$

where $\mathbf{1}_3 \in \mathbb{R}^{3\times 1}$ is a unit vector, ${}^w f^x_{k+1}$ and ${}^w f^y_{k+1} \in \mathbb{R}^{3\times 1}$ are respectively defined as

$${}^w f^x_{k+1} \triangleq \begin{bmatrix} {}^w f^x_{s_{i+1}} \\ {}^w f^x_{s_{i+2}} \\ {}^w f^x_{s_{i+3}} \end{bmatrix}, \; {}^w f^y_{k+1} \triangleq \begin{bmatrix} {}^w f^y_{s_{i+1}} \\ {}^w f^y_{s_{i+2}} \\ {}^w f^y_{s_{i+3}} \end{bmatrix}, \tag{3.80}$$

while the matrices $\mathbf{R}_x$ and $\mathbf{R}_y \in \mathbb{R}^{3\times 6}$ are defined as

$$\mathbf{R}_x \triangleq \begin{bmatrix} c\theta_{s_i} & -s\theta_{s_i} & 0 & 0 & 0 & 0 \\ c\theta_{s_i} & -s\theta_{s_i} & c\theta_{s_{i+1}} & -s\theta_{s_{i+1}} & 0 & 0 \\ c\theta_{s_i} & -s\theta_{s_i} & c\theta_{s_{i+1}} & -s\theta_{s_{i+1}} & c\theta_{s_{i+2}} & -s\theta_{s_{i+2}} \end{bmatrix},$$

$$\mathbf{R}_y \triangleq \begin{bmatrix} s\theta_{s_i} & c\theta_{s_i} & 0 & 0 & 0 & 0 \\ s\theta_{s_i} & c\theta_{s_i} & s\theta_{s_{i+1}} & c\theta_{s_{i+1}} & 0 & 0 \\ s\theta_{s_i} & c\theta_{s_i} & s\theta_{s_{i+1}} & c\theta_{s_{i+1}} & s\theta_{s_{i+2}} & c\theta_{s_{i+2}} \end{bmatrix},$$

with $s\theta_{s_i}$ and $c\theta_{s_i}$ standing for $sin\theta_{s_i}$ and $cos\theta_{s_i}$, respectively. The vector of relative steps positions $\triangle\mathbf{F}_{s_i}$ is given by

$$\triangle\mathbf{F}_{s_i} = \begin{bmatrix} f_{s_{i+1}}^{x\_s_i} & f_{s_{i+1}}^{y\_s_i} & f_{s_{i+2}}^{x\_s_{i+1}} & f_{s_{i+2}}^{y\_s_{i+1}} & f_{s_{i+3}}^{x\_s_{i+2}} & f_{s_{i+3}}^{y\_s_{i+2}} \end{bmatrix}^T \tag{3.81}$$

Assuming that the prediction period N_pT coincides with the period of two walking cycles, this means that we only predict three steps ahead of the current step. Accounting now for the feet orientations, Equations (3.79)-(3.81) can be substituted in (3.67) to give in compact form the expressions of the footsteps positions and orientations over the prediction horizon N_p as

$$\begin{cases} {}^w\mathbf{\Theta}_{k+1}^{ref} & = V_{k+1}^c\, {}^w\theta_k^c + V_{k+1}^f\, {}^w\boldsymbol{\theta}_{k+1}^f \\ {}^w\boldsymbol{F}_{k+1}^{x\_ref} & = \left(V_{k+1}^c + V_{k+1}^f \mathbf{1}_3\right) {}^w f_k^x + V_{k+1}^f \cdot \mathbf{R}_x \cdot \triangle\mathbf{F}_{s_i}\ . \\ {}^w\boldsymbol{F}_{k+1}^{y\_ref} & = \left(V_{k+1}^c + V_{k+1}^f \mathbf{1}_3\right) {}^w f_k^y + V_{k+1}^f \cdot \mathbf{R}_y \cdot \triangle\mathbf{F}_{s_i} \end{cases} \tag{3.82}$$

Note that ${}^w\mathbf{\Theta}_{k+1}^{ref}$ in (3.82) does not describe the orientation of robot's trunk over N_p, but the discrete orientations of the stance feet. To describe the continuous orientation of the robot's trunk, a set of equations of the form (3.68)-(3.70), can thus be used [27]. A block diagram of this scheme is given in Figure 3.8.

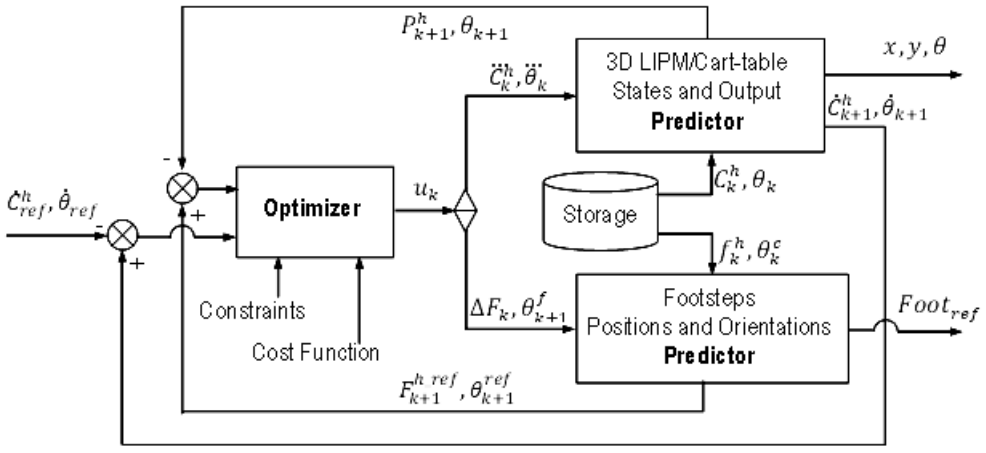


Figure 3.8: Reactive omnidirectional velocity follower pattern generator

3.3.3.1 Objective function

The footsteps placements and their orientations can now be generated automatically by using Equation (3.82) in an optimization scheme of the form (3.71) augmented with a cost function related to the orientation. The objective is now to find the *best* vectors $\triangle\mathbf{F}_{s_i}$ and ${}^w\boldsymbol{\theta}_{k+1}^f$, respectively, of footsteps relative positions and orientations such that errors between given translational

and rotational velocities and the robot's velocities over the prediction horizon is minimized while satisfying walking constraints. Thus, the optimization problem can be written as

$$\min_{u_k} \{g(x,y) + g(\theta)\} \tag{3.83}$$

where $g(x,y)$ and $g(\theta)$ are defined as

$$\begin{cases} g(x,y) & \triangleq \sum\limits_{h=x,y} \left\{ \frac{\mu}{2} \left\| \dddot{C}^h_{k+1} \right\|^2 + \frac{\sigma}{2} \left\| \dot{C}^h_{k+1} - \dot{C}^h_{ref} \right\|^2 \right. \\ & \left. + \frac{\varepsilon}{2} \left\| EC^h_{k+1} - \dot{C}^h_{ref} \right\|^2 + \frac{\kappa}{2} \left\| P^h_{k+1} - {}^w F^{h\_ref}_{k+1} \right\|^2 \right\} \\ g(\theta) & \triangleq \left\{ \frac{\mu}{2} \left\| \dddot{\theta}_{k+1} \right\|^2 + \frac{\sigma}{2} \left\| \dot{\theta}_{k+1} - \dot{\Theta}_{ref} \right\|^2 \right. \\ & \left. + \frac{\varepsilon}{2} \left\| E\theta_{k+1} - \dot{\Theta}_{ref} \right\|^2 + \frac{\kappa}{2} \left\| \theta_{k+1} - {}^w \Theta^{ref}_{k+1} \right\|^2 \right\} \end{cases} \tag{3.84}$$

and the vector u_k is given by

$$u_k = \begin{bmatrix} \dddot{C}^x_{k+1} & \dddot{C}^y_{k+1} & \triangle \mathbf{F}_{s_i} & \ddot{\theta}_{k+1} & \boldsymbol{\theta}^f_{k+1} \end{bmatrix}^T \tag{3.85}$$

In canonical form, the optimization (3.83) can be expressed as

$$\min_{u_k} \frac{1}{2} \cdot u_k^T \mathbf{Q_k} u_k + \mathbf{p_k}^T u_k \tag{3.86}$$

with

$$\mathbf{Q_k} = \begin{bmatrix} Q_{x11} & 0 & Q_{x12} & 0 \\ 0 & Q_{y11} & Q_{y12} & 0 \\ Q_{x21} & Q_{y21} & Q_{xy22} & 0 \\ 0 & 0 & 0 & Q_\theta \end{bmatrix} \tag{3.87}$$

$$\mathbf{p_k} = \begin{bmatrix} p_{k\_\ddot{c}^x} & p_{k\_\ddot{c}^y} & p_{k\_f^{x+y}} & p_{k\_\ddot{\theta}} & p_{k\_f^\theta} \end{bmatrix}^T \tag{3.88}$$

Using the system of Equation (3.82) instead of (3.67), the elements of $\mathbf{Q_k}$ and $\mathbf{P_k}$ can be easily deduced from those computed in [27], from where the notation used above has been adapted. Nevertheless, the formulation has to conform with the new vector u_k given by Equation (3.85). Hence, we obtain the matrix $\mathbf{Q_k}$ as

$$\begin{cases} Q_{h11} & = \alpha I + \beta U_{\dot{h}}^T U_{\dot{h}} + \gamma U_{p_h}^T U_{p_h} + \varepsilon U_h^T E^T E U_h \\ Q_{h12} & = -\gamma U_{p_h}^T V^f_{k+1} \mathbf{R}_h \\ Q_{h21} & = -\gamma (V^f_{k+1} \mathbf{R}_h)^T U_{p_h} \qquad \text{with } h = x, y \\ Q_{h22} & = \gamma (V^f_{k+1} \mathbf{R}_h)^T (V^f_{k+1} \mathbf{R}_h) \\ Q_{xy22} & \triangleq Q_{x22} + Q_{y22} \end{cases} \tag{3.89}$$

$$Q_\theta = \begin{bmatrix} \alpha I + \beta U_{\dot{\theta}}^T U_{\dot{\theta}} + \gamma U_{p_\theta}^T U_{p_\theta} + \varepsilon U_\theta^T E^T E U_\theta & -\gamma U_{p_h}^T V_{k+1}^f \\ -\gamma (V_{k+1}^f)^T U_{p_\theta} & \gamma (V_{k+1}^f)^T V_{k+1}^f \end{bmatrix}, \tag{3.90}$$

and the vector $\mathbf{p_k} \triangleq \begin{bmatrix} \mathbf{p}_{\mathbf{k}_{xy}}^T & \mathbf{p}_{\mathbf{k}\theta}^T \end{bmatrix}^T$, with

$$\mathbf{p}_{\mathbf{k}_{xy}} = \begin{bmatrix} \Big\{\beta U_{\dot{x}}^T (S_{\dot{x}} \hat{c}_k^x - \dot{C}_{ref}^x) + \varepsilon U_x^T E^T (E S_x \hat{c}_k^x - \dot{C}_{ref}^y) \\ \quad + \gamma U_{p_x}^T (S_{p_x} \hat{c}_k^x - (V_{k+1}^c + V_{k+1}^f \mathbf{1})^w f_k^x) \Big\} \\ \Big\{\beta U_{\dot{y}}^T (S_{\dot{y}} \hat{c}_k^y - \dot{C}_{ref}^y) + \varepsilon U_y^T E^T (E S_y \hat{c}_k^y - \dot{C}_{ref}^y) \\ \quad + \gamma U_{p_y}^T (S_{p_y} \hat{y}_k - (V_{k+1}^c + V_{k+1}^f \mathbf{1})^w f_k^y) \Big\} \\ -\gamma \Big\{ (V_{k+1}^f \mathbf{R}_x)^T (S_{p_x} \hat{c}_k^x - (V_{k+1}^c + V_{k+1}^f \mathbf{1})^w f_k^x) \\ \quad + (V_{k+1}^f \mathbf{R}_y)^T (S_{p_y} \hat{c}_k^y - (V_{k+1}^c + V_{k+1}^f \mathbf{1})^w f_k^y) \Big\} \end{bmatrix} \tag{3.91}$$

$$\mathbf{p}_{\mathbf{k}_\theta} = \begin{bmatrix} \Big\{\beta U_{\dot{\theta}}^T (S_{\dot{\theta}} \hat{\theta}_k - \dot{\Theta}_{ref}) + \varepsilon U_{pu}^T E^T (E S_{ps} \hat{\theta}_k - \dot{\Theta}_{ref}) \\ \quad + \gamma U_{zu}^T (S_{zs} \hat{\theta}_k - V_{k+1}^{c\ w} \theta_k^c) \Big\} \\ \Big\{ -\gamma V_{k+1}^{f\ T} (S_{zs} \hat{\theta}_k - V_{k+1}^{c\ w} \theta_k^c) \Big\} \end{bmatrix} \tag{3.92}$$

The optimization process will provide viable walking solutions only if the constraints invoked previously and well detailed in [27] are satisfied. However, the formulation of constraints on the CoP has to comply with (3.82) and adapted for the new u_k vector (3.85). These constraints are given in Appendix A.1.

Remark 1. Unlike in [23, 27, 19], where the footsteps are generated with respect to an absolute frame (static inertial frame), the footsteps in the proposed extension are instead generated relative to the previous ones. This has the advantage to transform a global problem in a local problem, offering as such possibilities of local linearization. This can be observed in (3.79) where the relative footsteps are linearly mapped to the future steps by the matrices $\mathbf{R}_x$ and $\mathbf{R}_y$, which are function of the orientation. Particularly, we emphasize on the orientation of the stance foot, which is discrete over the time, instead of the continuous orientation of the CoM or the trunk. Thus, the curvilinear trajectory of the robot due to the rotational motion is approximated by a piecewise linear function. For instance, describing the trajectory over two complete cycles has required only three future values of θ. Moreover, by assuming slow variations of the predicted values of θ between two sample instants during a step, their previously obtained values are used to compute $\mathbf{R}_x$ and $\mathbf{R}_y$ at each iteration. In this way, we have minimized the induced non-linearities and thereby addressed the challenge mentioned in [23, 27, 19]. Thus, this results in *a reactive omnidirectional walking pattern generator* solved as a linear problem.

Remark 2. Besides the automatic generation of footsteps positions, the 3D trajectories of the swing foot have also to be dynamically planned and executed. Indeed, before becoming stance

foot, and depending on the reference velocity, the future position of the swing foot on the ground is continuously adapted by the walking algorithm. In order to generate continuous swing foot trajectories in position, velocity, and acceleration and also to ensure zero velocity at the takeoff and landing of the foot, current and predicted footsteps can then be used, for instance, with polynomial interpolations (see Appendix A.3 for more details).

3.4 Simulations of the Reactive Omnidirectional Walking Pattern Generator

This section presents simulation results validating the proposed pattern generator, which solves concurrently translations and rotation to generate stable humanoid's trajectories while following given reference velocities.

The validation was carried out on a simulator we wrote in Matlab for the purpose of this research. It uses the parameters of the humanoid NAO [144] which is presented and explicitly modeled in chapter 5. This simulator couples the proposed reactive omnidirectional walking pattern generator (ROWPG) and dynamic planner of the swing foot trajectories with the model of NAO robot, whose links are represented by wire-frames. More specifically, the walking algorithm takes the desired CoM velocity as input and provide the stance foot positions and orientations, the trajectories of the CoM and swing foot as outputs. Using these output trajectories, the inverse kinematics model of NAO robot computes the joints values to be executed. This allows a 3D visualization of the robot's motions.

For the results presented in the sequel, the LMPC algorithm of the pattern generator had a sampling time T of 100 ms, a total step time (SSP + DSP) t_s of 800 ms and the cost function's weights (Equation 3.84) $\mu = 10^{-8}$, $\sigma = 0.1$, $\varepsilon = 1.0$ and $\kappa = 1.8$. The height of the CoM was set to 0.275 m. For the locomotion, only the lower part of the 3D wire-frame model will be shown.

In order to qualitatively assess the generated walking trajectories, the following variables will be plotted and analyzed:

- the X and Y trajectories of the CoM and the corresponding trajectories of the ZMP,
- the CoM's reference and instantaneous velocities,
- the 2D poses of the generated footsteps and their corresponding support polygons,
- the swing foot trajectories relative to the stance foot,
- finally, 3D sequences of the walking motion in order to visualize how the actual robot will behave when executing the generated trajectories.

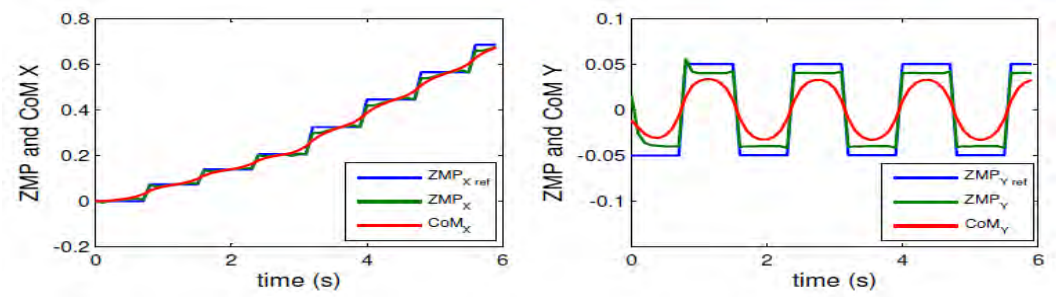
Given that the proposed algorithm couples the longitudinal and the lateral translational motions, it is important to check that pure translation in one direction or the other can be followed normally. Hence, we will start by simple translations and gradually move up to combining translations and rotations.

3.4.1 Translations

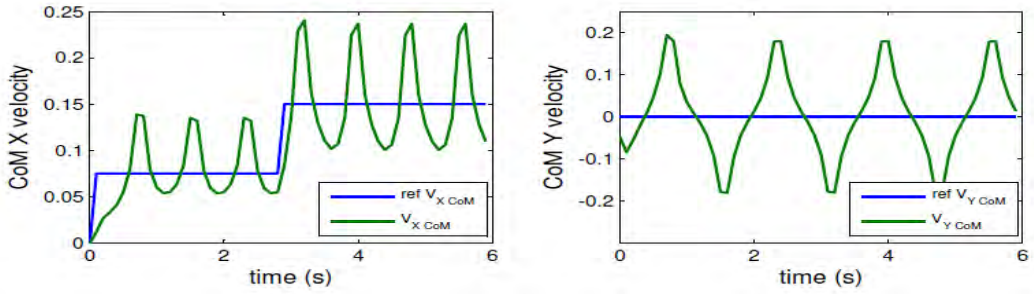
3.4.1.1 Longitudinal Translation

Consider a case where the robot starts at the origin of axes in double support phase with the X reference velocity stepped from 0.0 m/s to 0.075 m/s. After three footsteps, the reference velocity is stepped from 0.075 m/s to 0.15 m/s.

The walking trajectories generated by the ROWPG for this motion are shown in Figure 3.9, where can be seen the trajectories of the reference and actual ZMP, and the CoM reference and instantaneous velocities.



(a) X and Y Reference and Actual ZMP trajectories and CoM trajectory during forward walking



(b) X and Y Reference and Actual velocities of the Humanoid CoM for forward walking

Figure 3.9: Desired and actual CoM and ZMP's trajectories generated for longitudinal input velocity

It can be observed that the average X CoM's velocity follows its reference velocity, while its Y counter part stays zero. Regarding gait stability, though the actual ZMP does not track perfectly its reference (auto-generated footsteps pose), the humanoid is stable. Indeed, the actual ZMP stays within the support polygon as confirmed in Figure 3.10 representing the 2D the positions

of the left (magenta) and right (black) footsteps, the trajectories of the generated reference ZMP, and the CoM trajectory with respect to a fixed reference frame.

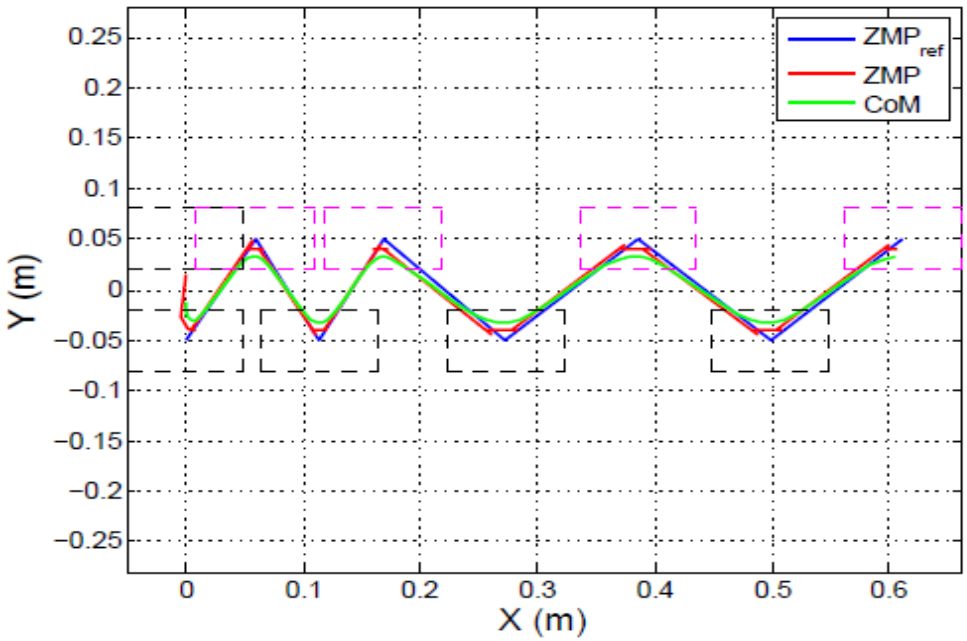


Figure 3.10: Footsteps placement, ZMP and CoM trajectories in forward walking

The swing foot trajectories relative to the stance foot are illustrated in Figure 3.11 where, from negative to positive values, ΔX of the swing foot varies similarly to observed humans forward walking.

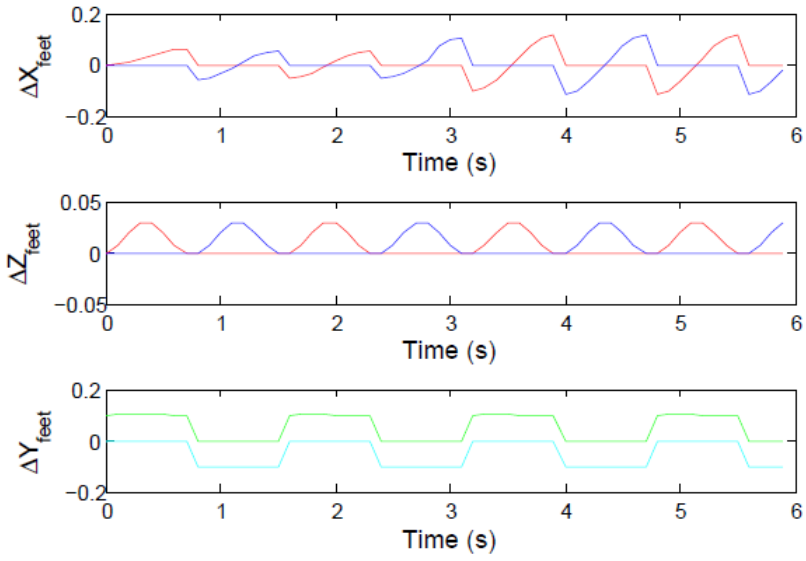


Figure 3.11: Swing foot trajectories in forward walking

Finally, on Figure 3.12 we provide a 3D representation of the walking motion, depicting

the sequence of each step and also showing the swing foot and the center of mass trajectories, respectively in red for the right foot, in blue for the left foot and in green for the center of mass.

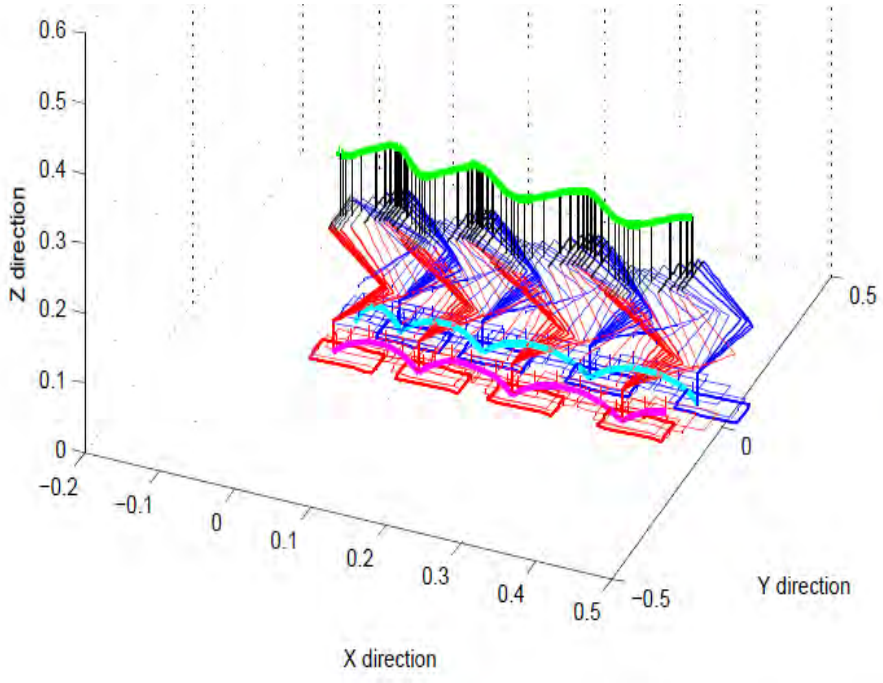


Figure 3.12: 3D trajectories and sequence of forward walking motion

3.4.1.2 Lateral Translation

Consider now a pure translation in the Y direction. The robot starts at the origin with a reference velocity stepped from $0\,m/s$ to $0.05\,m/s$, after 4.0 seconds, the velocity is stepped again from $0.05\,m/s$ to $0.1\,m/s$ and finally after 4 steps, the velocity is increased to $0.15\,m/s$.

Figure 3.13 illustrates the time evolution of the CoM and ZMP coordinates. It can be seen that no motion is happening in the X direction as also confirmed in Figure 3.14 showing the 2D trajectory of the CoM, the desired and actual ZMP and their associated support polygons.
In Y direction, however, it can be observed that the average CoM velocity follows well the reference velocity ($0.5\,m/s$) for the first 4 seconds; when the velocity is set to $0.1\,m/s$, the results is still acceptable (from $4\,s$ to $7\,s$) but the ZMP of the right foot varies around the reference ZMP. When the velocity reached $0.15\,m/s$, the robot cannot track such a velocity due to the walking constraints, the ZMP of the right foot now stays completely below its reference with a constant value indication the activeness of constraints.

The swing foot trajectories are represented in Figure 3.15. There is no trajectory planned in the X direction, meanwhile the Z trajectory repeats the same pattern, explained by the fact that each foot is planned to be lifted up to a prescribed maximum height.

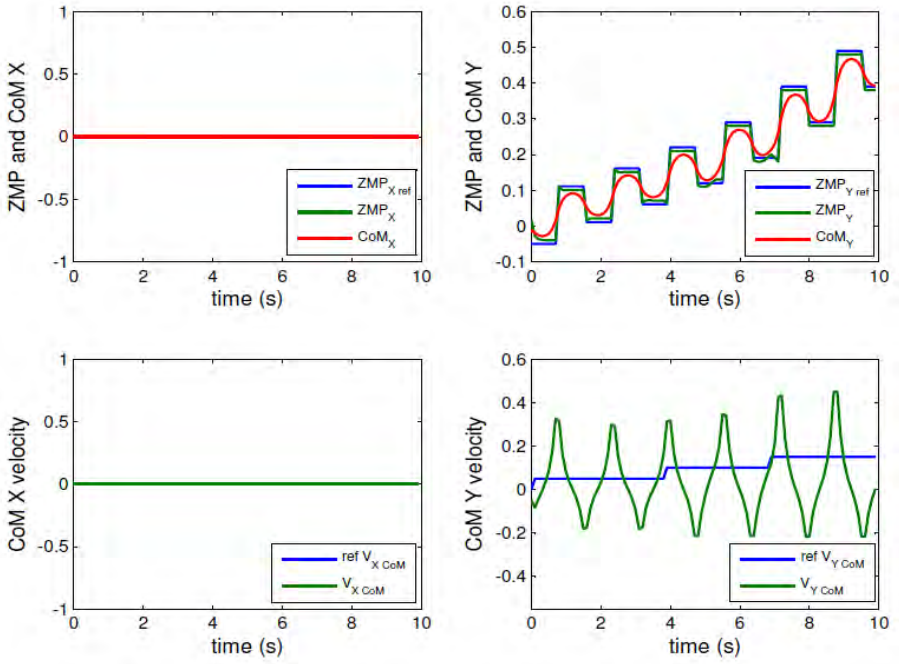


Figure 3.13: Lateral walking motion

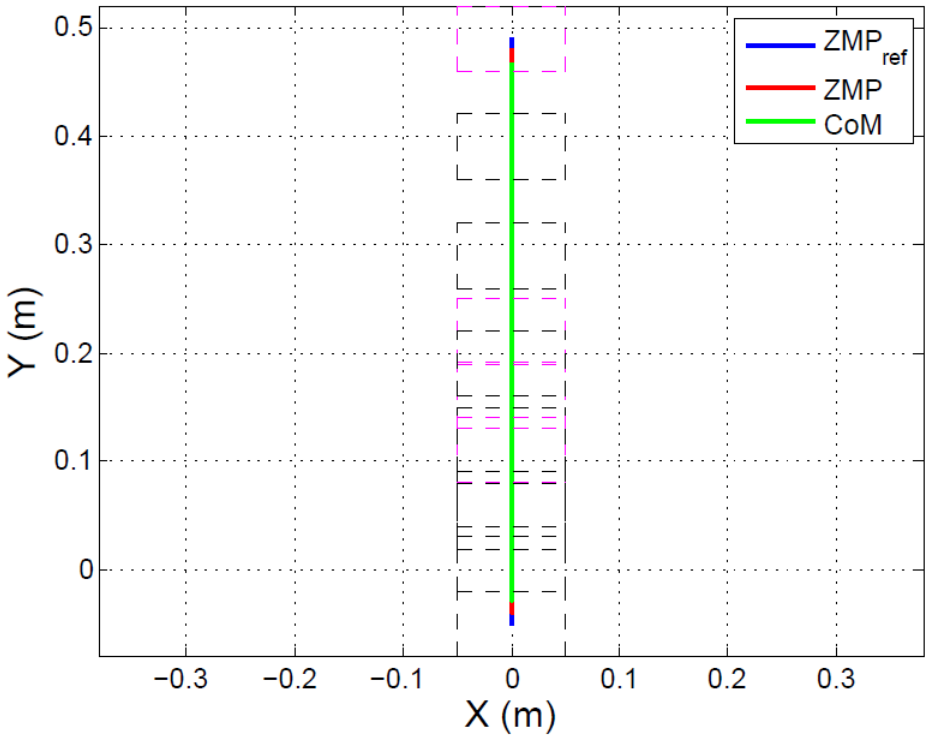


Figure 3.14: Footsteps placement, ZMP and CoM trajectories during lateral walking motion

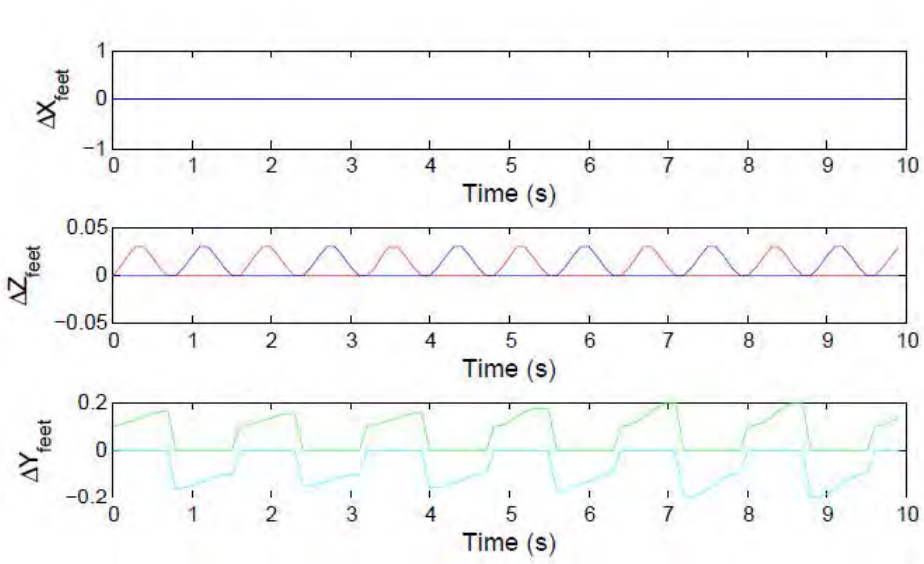


Figure 3.15: Swing foot trajectory during lateral walking motion

On the Δy plot, where a zero value corresponds to the stance foot, Figure 3.15 tells us that the robot initially moved with a right stance foot. The left foot, from the initial inter-feet distance, moves towards the robot's left hand side, the direction of the reference velocity. When the left swing foot reaches the ground, the right foot becomes the new swing foot and starts moving, relatively to the left stance foot, from a negative position up to the minimum inter-feet distance allowed by walking constraints. Hence, the cycle goes on.

3.4.1.3 Combined Longitudinal and Lateral Translations

Consider now a combination of X and Y translations. Starting at the origin of axes with a X velocity of $0.10\,m/s$ and a Y velocity of $+0.05\,m/s$, the robot walks for 5 seconds before seeing its Y velocity switched to $-0.10\,m/s$ as illustrated in Figure 3.16.

On this Figure, one can noticed that at the moment Y velocity was switched from positive to negative value, the robot was standing on its right foot. Given that crossing legs is not allowed, the robot had first to change its support foot before moving with a negative Y velocity.

The sequence of the robot's motion and its 3D CoM and feet trajectories as they occur are shown in Figure 3.17. The forward progression of the robot, and its constant orientation while moving to the left and then to the right after $0.5\,m$ (5 seconds with a $v_x = 0.10m$) in the X direction can be clearly seen.

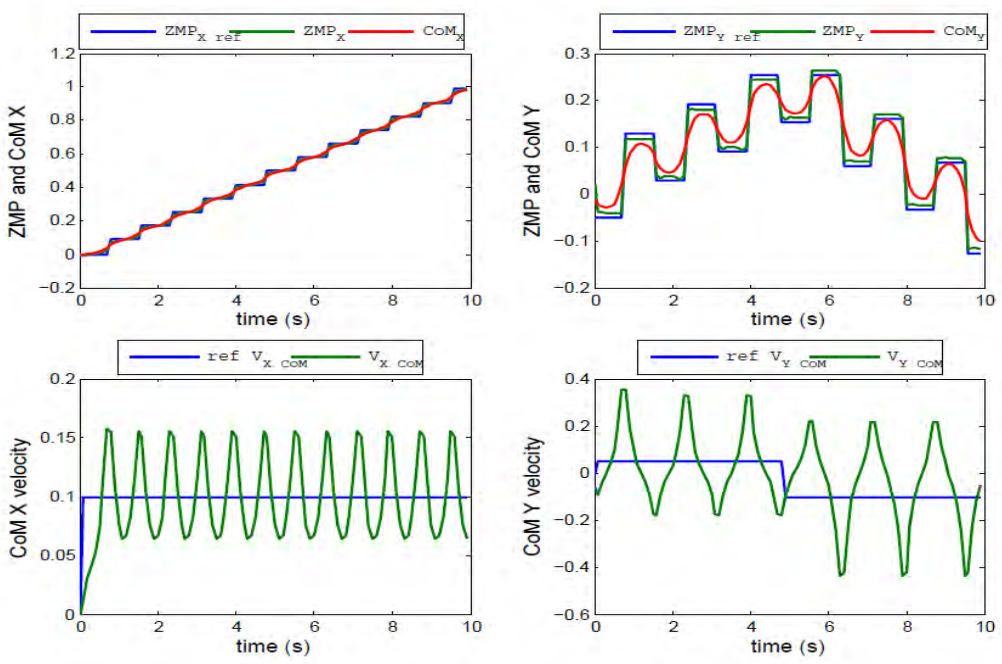


Figure 3.16: Translation XY

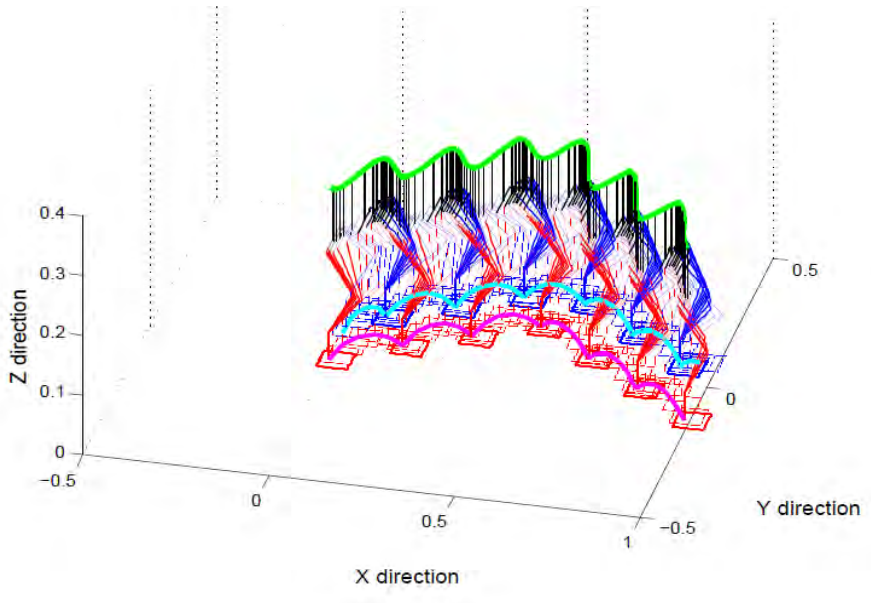


Figure 3.17: 3D trajectories and sequences of motion during combined translations

How stable the motion is can be appreciated in Figure 3.18, depicting the trajectories of the ZMP and the projection of the CoM on the ground. It can be observed that the actual ZMP moves from the back to the front of its prescribed value, but stays within the support polygon. This clearly tells that the generated trajectories are stable and allow to follow reference velocities.

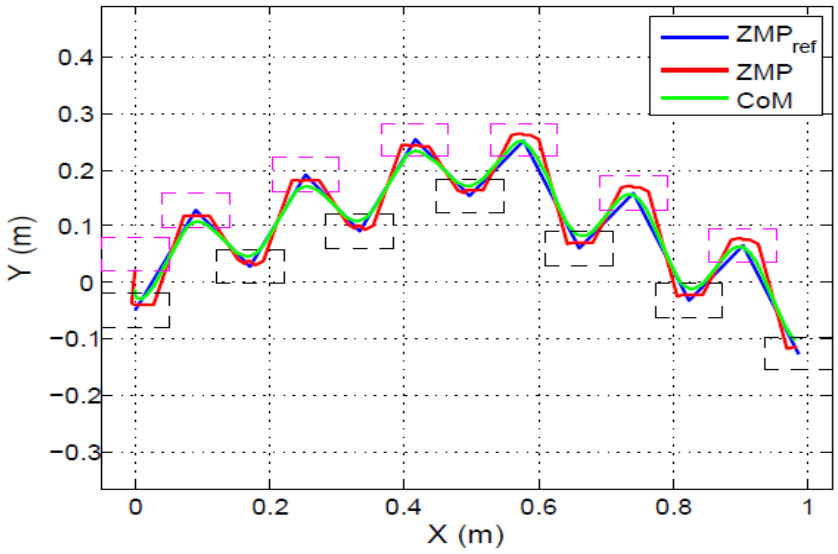


Figure 3.18: Footsteps placements, ZMP and CoM trajectories during X-Y translations

3.4.2 Rotations

Let us consider now a pure rotational motion. The robot is located at the origin, the velocities in X and Y directions are kept at zero while the reference rotational velocity ω_z is stepped first from $0\,rad/s$ to $0.15\,rad/s$, after 4 seconds it is stepped again from $0.15\,rad/s$ to $0.3\,rad/s$ and finally from $0.3\,rad/s$ to $0.45\,rad/s$ after 3 seconds.

The profile of this reference velocity and the actual velocity of the CoM are represented in Figure 3.19. There can also be seen the generated reference orientation of stance foot and the actual orientation of the CoM.

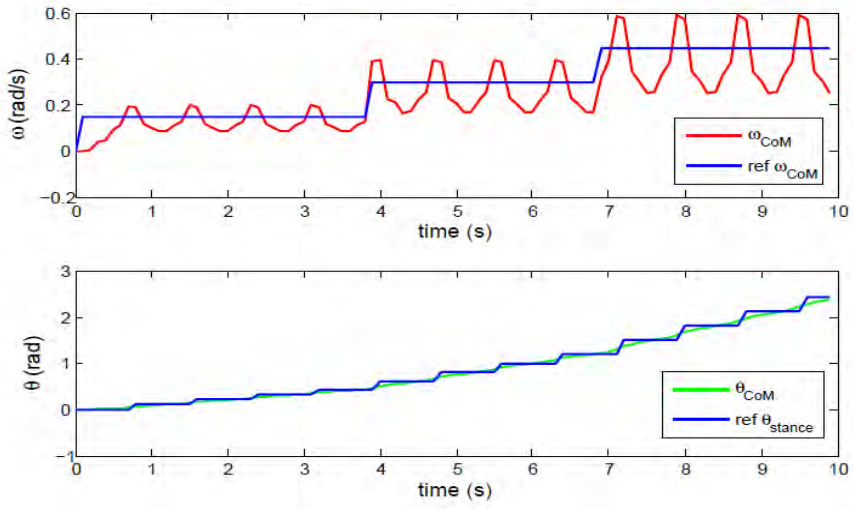


Figure 3.19: Reference rotational velocity, and CoM and stance foot orientations

The X and Y trajectories of the ZMP generated in order to follows such a velocity profile are shown in Figure 3.20.

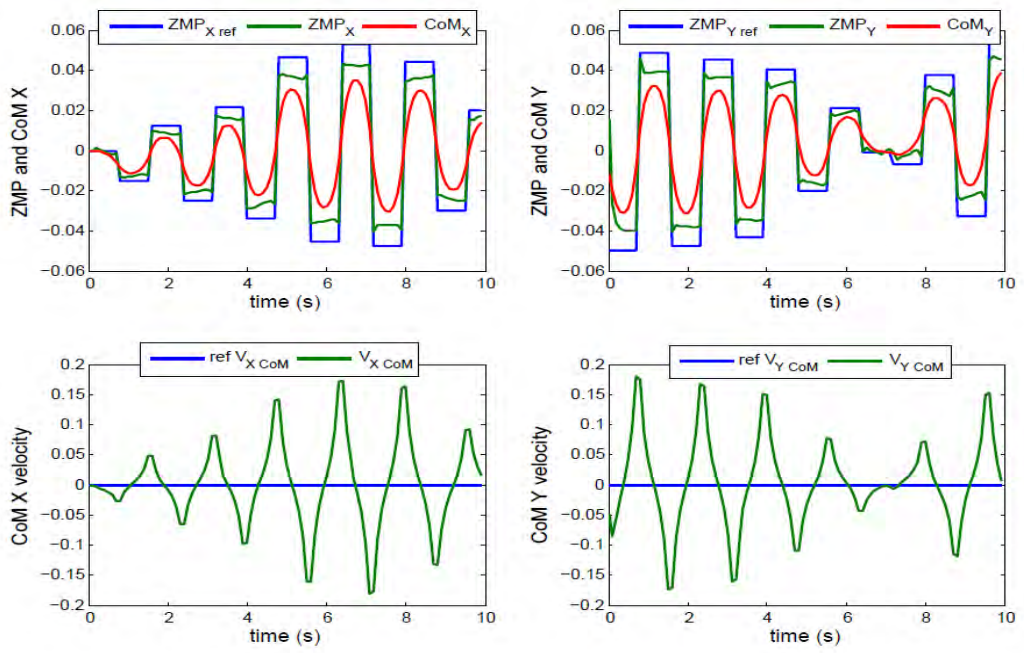


Figure 3.20: ZMP and CoM trajectories during pure rotational motion

The average values of the X and Y CoM's velocity are zero, showing clearly that there is no global translation of the robot. The observed CoM's motion is performed to keep the balance of the robot while changing successively its support foot during the rotation. Also, the observed oscillations between negative and positive values of the X trajectory of the ZMP corresponds to successive backward and forward steps with left foot and right foot respectively. In this particular case, the robot moves in an anti-clockwise direction. Starting with a right stance foot, the robot moves its left foot backward while changing its orientation in accordance with the reference velocity. Once the left foot on the ground, the right foot becomes the swing foot, which now moves forward while also changing its orientation.

On Figure 3.21, we show the footsteps as they are realized on the floor. It can be observed clearly that the ZMP in red doesn't track perfectly its reference in blue, but it stays within the support polygon. It can also be observed three different spacing in between the steps, which correspond to the three velocity set-points. This fact is also confirmed by the swing foot trajectories represented on Figure 3.22, where changes of profile on the Δx plot can be seen at $4\,s$ and at $7\,s$. Constant velocity should have yielded regular spacings between footsteps, similar to the one represented in Figure 3.23 and corresponding to a constant velocity of 0.2 rad/s.

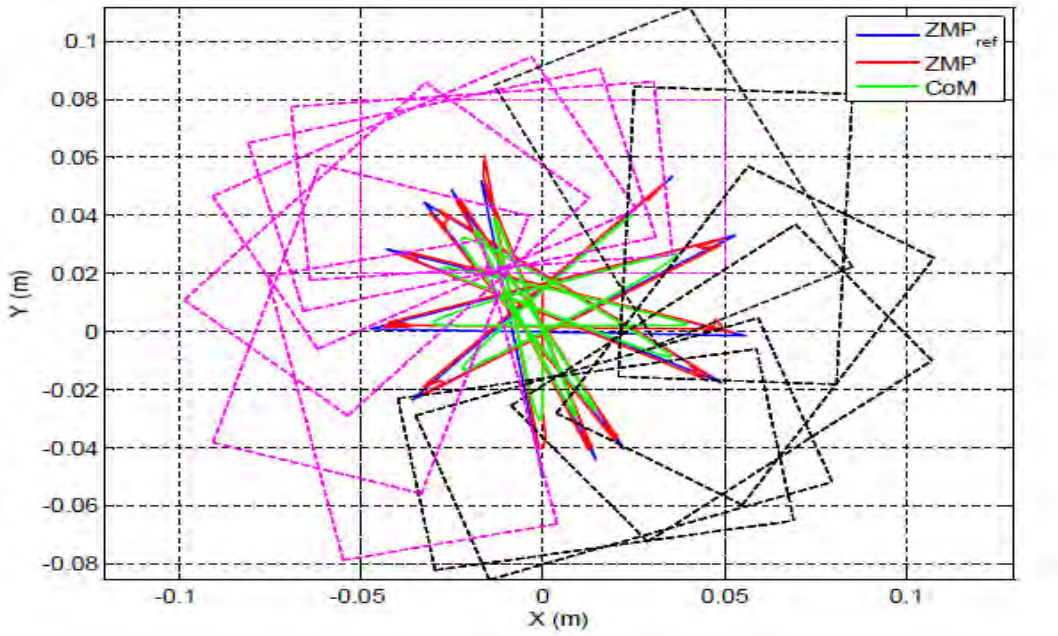


Figure 3.21: Footsteps poses during rotational motion

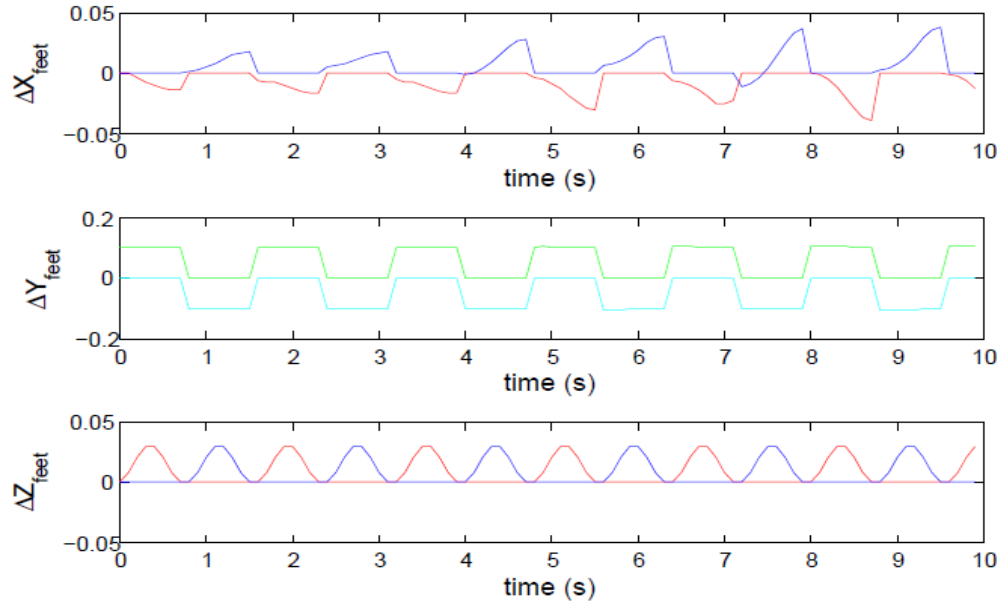


Figure 3.22: Swing foot trajectories in pure rotational motion

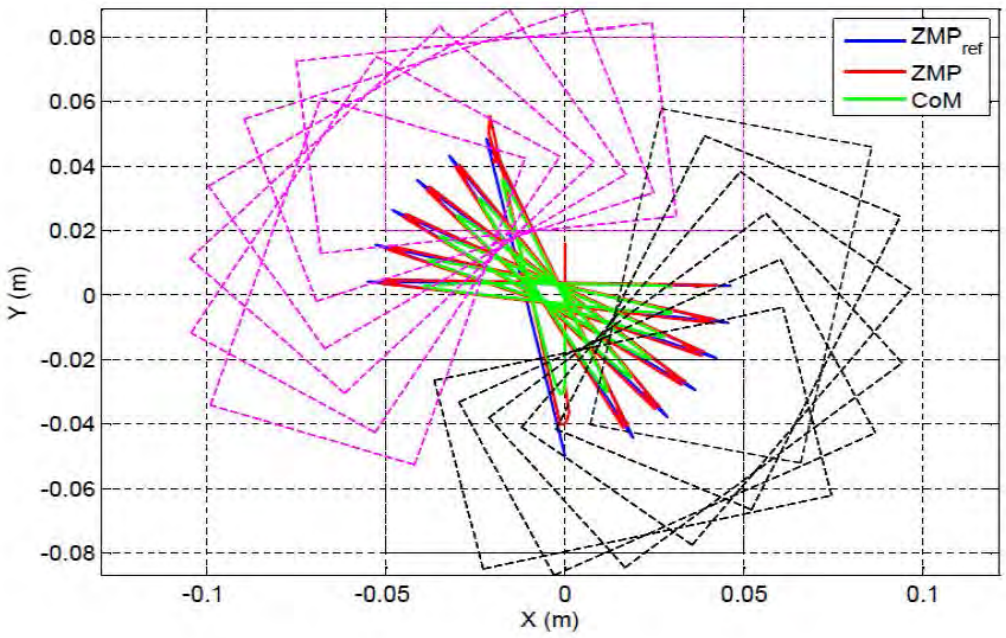


Figure 3.23: Regular spacing between footsteps during constant rotational motion

3.4.3 Combined Translations and Rotation

Consider now a walking motion combining both translations and rotation. In this simulation, the velocity set-points are defined relative to the robot such that the forward and backward motions correspond to the x direction of the robot and the lateral motions to the robot's y direction.

The robot starts walking from the origin with a relative x velocity of $+0.075\,m/s$ and an angular velocity of $-0.20\,m/s$. After eight seconds, these velocities are set to zero except the y velocity which is stepped to $+0.10\,m/s$. Five seconds later, the y velocity is reduced to $+0.075\,m/s$, while the angular velocity is stepped to $+0.02\,rad/s$. At the 20^{th} second, the relative x velocity is set to $-0.075\,m/s$ while the y and the angular velocity are set to zero. Also, from the 25^{th} to the 27^{th} second, the robot performs a pure rotation with $+0.15\,rad/s$. Finally, the robot combines, for five seconds, a forward motion at $+0.10\,m/s$ with an angular velocity $+0.30\,rad/s$.

The rotational motion being our main feature, in Figure 3.24 we show the profile of the angular velocity and the discrete orientation of the stance foot and the associated continuous orientation of the robot's CoM. The ZMP and CoM's trajectories, relative to the world frame, are shown in Figure 3.25. A good indication of the humanoid's stability and its displacement are provided in Figure 3.26, where the footsteps positions on the floor and the trajectories of the ZMP and CoM are represented.

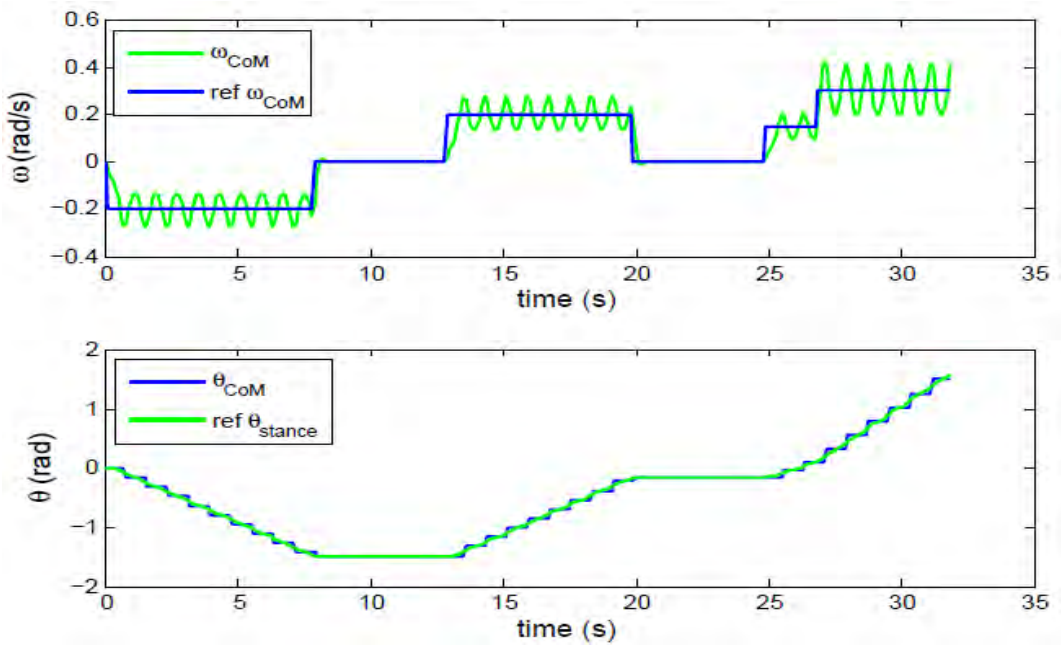


Figure 3.24: Profile of rotational velocity in a combined translational and rotational motion

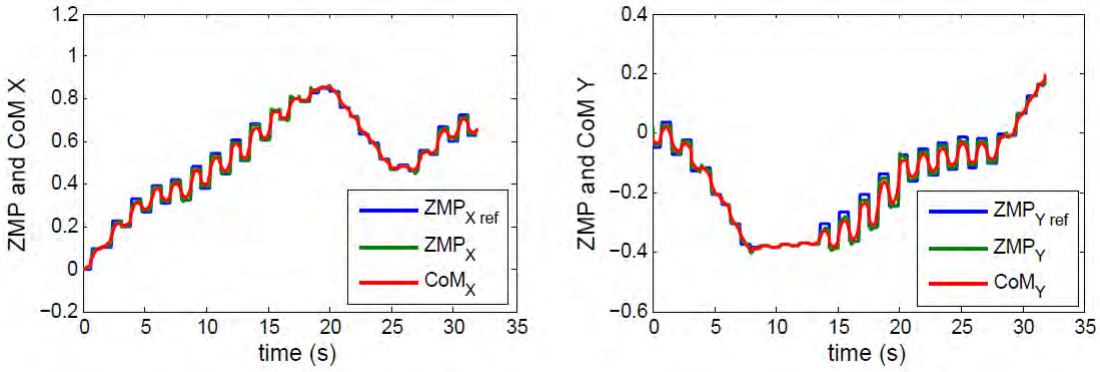


Figure 3.25: ZMP and CoM trajectories during combined translation and rotation

Finally, in Figure 3.27, we give the 3D sequence corresponding to this complex walking motion. Besides the CoM's trajectory in green, it shows the feet trajectories (magenta and sky blue) as realized without feet crossing or collisions (important for the actual implementation).

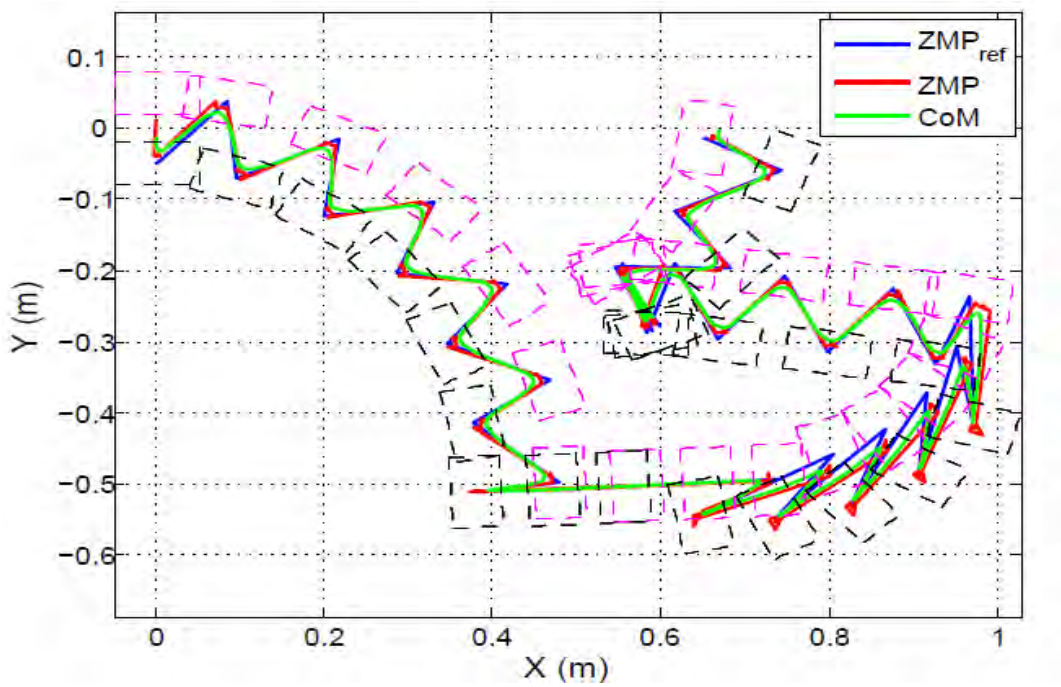


Figure 3.26: Footsteps placements, ZMP and CoM trajectories for combined translation and rotation

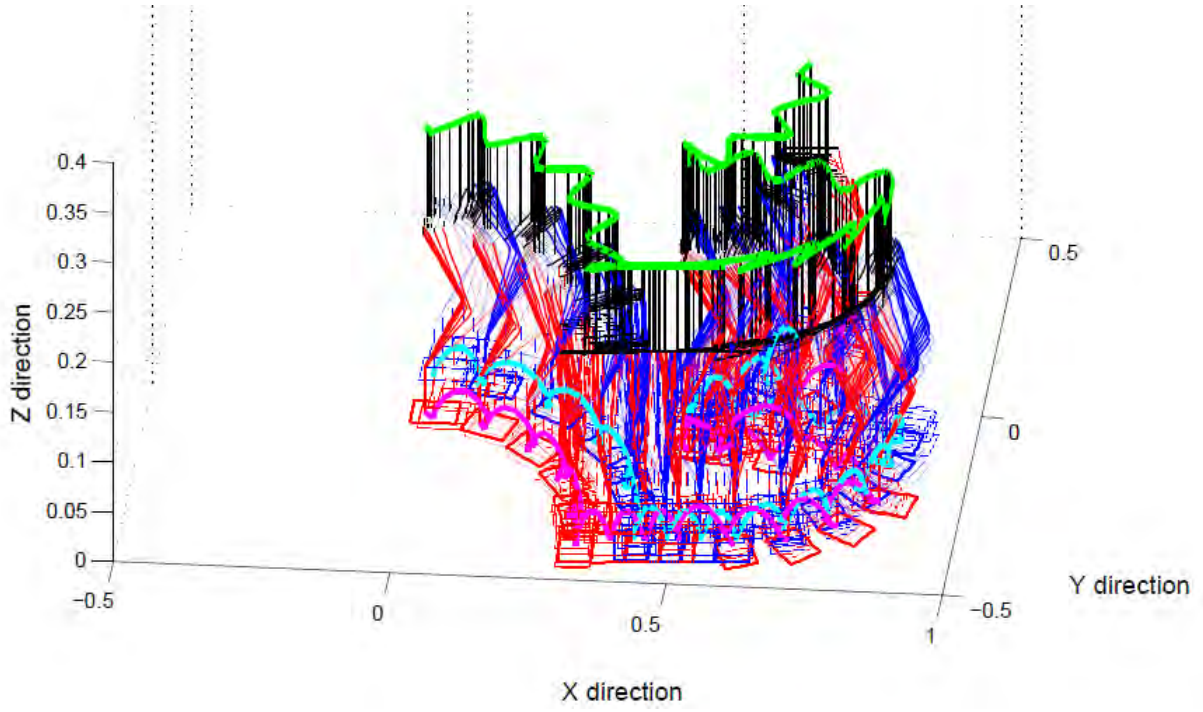


Figure 3.27: 3D sequence of humanoid trajectory for combined translational and rotational motion

Remark. In view of the results presented, one may argue about the ZMP tracking, especially in Y direction. In response, we recall that the goal is not to have a precise and fixed ZMP during a step, but to let it move as long as it stays in the stability region and the robot follows to its best the reference velocity. Nevertheless, the tracking performance can be improved by tuning the weight factors in the LMPC's cost function. Also, when the reference velocities change abruptly, filtering the set-points could also improve the trajectories by smoothing all step inputs and thereby reducing the jerk.

3.5 Conclusion

In this chapter, using Lagrange and D'Alambert's principle, we have derived the model of a biped humanoid robot, which is characterized by a hybrid dynamics subject to different constraints. We have discussed the control problem to ensure postural balance and stable dynamic locomotion. In that respect, we have presented the concept of ZMP and introduced LMPC based pattern generators using simplified dynamic models of humanoid robot.

We stressed particularly on a reactive pattern generator able to generate automatically footsteps positions from reference velocities. We then proposed an extension to footsteps orientations in order to overcome their predetermination. As a result, we have obtained a reactive omnidirectional pattern generator, which has been validated through simulations. Indeed, without any planning of footsteps neither in position nor in orientation, from a given reference velocity, this pattern generator computes concurrently and in real-time optimal footsteps positions and their orientations necessary to follow the prescribed velocity and complying with the different walking constraints. Additionally, it generates the swing foot trajectories which can be used in conjunction with the inverse kinematics to compute the joints trajectories to be executed.

The simulation results confirmed that the proposed pattern generator is able to generate stable walking trajectories from different kinds of velocity demands. These velocity set-points may result from high-level control structure as we will see in the next chapter, where a visual servo controller generates reference velocities to drive a humanoid robot in order to perform given tasks.

Chapter 4

Visual Servoing on Humanoid Robot

This chapter, partially published in [145], presents a general approach to visual servoing of humanoid robots. In particular, the proposed approach handles concurrently upper body and lower body motions, thereby resulting in whole body motions directly driven by a visual servo controller. In the first section, a kinematic analysis is carried out to find a task Jacobian compatible with the robot's bipedal structure and its related locomotion constraints. Then follows the formulation of visual based whole body motion, where the redundancy problem, the integration of the walking motion and the compensation of tasks interactions are discussed and addressed. Finally, the proposed formulation is applied to positioning, tracking and grasping tasks.

4.1 Kinematic Modeling

Consider a humanoid robot, for instance, in single support phase. The robot can be modeled as a tree-like kinematic chain (leg joints + upper part joints) whose base is the stance foot and the end-effector could be either the head or one of the two hands as illustrated in Figure 4.1.

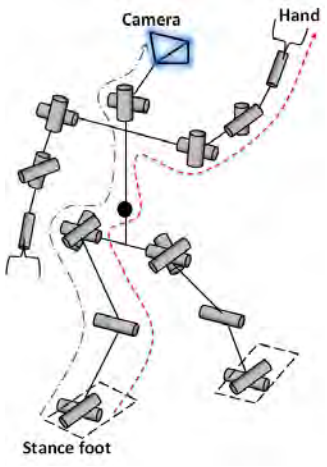


Figure 4.1: Humanoid's whole body chains

Applying visual servoing to a humanoid robot requires the computation of the task Jacobian, which itself depends on the robot Jacobian. To that end, let any end-effector configuration be described by a vector $\boldsymbol{r}_e = f(\mathbf{q})$, where $\mathbf{q}$ represents the joint configuration vector.

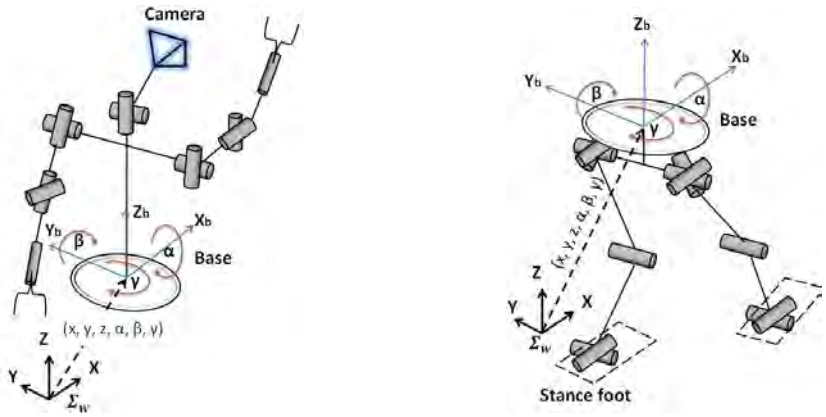
4.1.1 Task Jacobian: Inconsistent Approach

Using forward kinematics, it is known that the time variation of the end-effector configuration, $\dot{\boldsymbol{r}}_e$, will be related to the joint velocities by the Jacobian of the considered serial chain [124, 130, 126]. Thinking conversely, one might compute the joint velocities required to perform a task function of $\dot{\boldsymbol{r}}_e$ by using the (pseudo) inverse Jacobian of the considered chain. Although valid for a fixed base robot manipulator and to some extent for a wheeled robot, applying the latter approach on a humanoid robot may result in joints values not satisfying the postural balance and gait stability.

Indeed, this approach implicitly assumes continuous motion in the joints space with a fixed base on the ground (a support foot glued on the ground); whereas a biped robot is required to renew its support in order to keep balance and stability during its locomotion [16], thus giving rise to alternating support phases between the legs. Therefore, the modeling approach presented above and supposed to provide joints velocities for a given task has to be reconsidered in order to account for balance and stability.

4.1.2 Task Jacobian: Consistent Approach

Let us consider the humanoid's upper body as shown in Figure 4.2a. It undergoes, in a continuous manner, three translations and three rotations. We propose to study its kinematics by making first an abstraction on how these translations and rotations are produced and afterward we will relate these motions to the lower body realizing the locomotion. As a result, the overall model can be simplified; any end-effector of the upper body (head or the hands) can be considered as serial manipulator, with joints vector $\mathbf{q}_m$, fixed on a moving base.



(a) Upper body part (b) Lower body part

Figure 4.2: Reduced configuration model of humanoid robot

4.1.2.1 Humanoid's Upper Body Kinematics

Let the configuration of that base, with respect to an inertial reference frame $\mathcal{F}_w$, be described by the vector $\mathbf{r}_b \in SE(3)$ such that $\mathbf{r}_b = g(\mathbf{q}_b)$, with $\mathbf{q}_b = [\ \mathbf{t}_b^T \quad \boldsymbol{\phi}_b^T\ ]^T$. As in Section 3.1.2, let us also define $\mathbf{t}_b \triangleq [\ x_b \quad y_b \quad z_b\ ]^T$ and $\boldsymbol{\phi}_b^T \triangleq [\ \alpha_b \quad \beta_b \quad \gamma_b\ ]^T$ to represent the base's translation and orientation (parametrized here by Euler angle of type ZYX) of $\mathcal{F}_B$ relative to $\mathcal{F}_W$, respectively. With these considerations, the kinematic screw of the end-effector with respect to $\mathcal{F}_w$ can be related to the joints velocities by differentiating $\boldsymbol{r}_e = f(\mathbf{q})$ as follows

$$ {}^w\dot{\boldsymbol{r}}_e = \frac{\partial f}{\partial \mathbf{q}_m}\dot{\mathbf{q}}_m + \frac{\partial f}{\partial \mathbf{q}_b}\dot{\mathbf{q}}_b $$

which can be rewritten as

$$ {}^w\dot{\boldsymbol{r}}_e = \begin{bmatrix} \dfrac{\partial f}{\partial \mathbf{q}_m} & \dfrac{\partial f}{\partial \mathbf{q}_b} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_m \\ \dot{\mathbf{q}}_b \end{bmatrix} = {}^w\mathbf{J}_{mb}\,\dot{\mathbf{q}}_{mb} \tag{4.1} $$

where ${}^w\dot{\boldsymbol{r}}_e = [{}^w\boldsymbol{\upsilon}_e\ {}^w\boldsymbol{\omega}_e]^T \in se_3$, with ${}^w\boldsymbol{\upsilon}_e$ and ${}^w\boldsymbol{\omega}_e$ denoting the translational and rotational end-effector velocities, respectively. ${}^w\mathbf{J}_{mb}$ is the Jacobian of the considered end-effector in the inertial frame $\mathcal{F}_w$ with

$$ \dot{\mathbf{q}}_{mb} = \begin{bmatrix} \dot{\mathbf{q}}_m^T & \dot{\mathbf{q}}_b^T \end{bmatrix}^T \tag{4.2} $$

Now, we need to determine the expression of ${}^w\mathbf{J}_{mb}$. To that end, using the kinematics of rigid body, let us write the end-effector velocity as follows

$$ \begin{cases} {}^w\boldsymbol{\upsilon}_e &= {}^w\mathbf{R}_b\,{}^b\boldsymbol{\upsilon}_e + {}^w\boldsymbol{\upsilon}_b + {}^w\boldsymbol{\omega}_b \times {}^w\mathbf{R}_b\,{}^b\mathbf{t}_{re} \\ {}^w\boldsymbol{\omega}_e &= {}^w\mathbf{R}_b\,{}^b\boldsymbol{\omega}_e + {}^w\boldsymbol{\omega}_b \end{cases} \tag{4.3} $$

where ${}^w\mathbf{R}_b$ is the rotation matrix from the base frame $\mathcal{F}_b$ to the frame $\mathcal{F}_w$, ${}^b\boldsymbol{\upsilon}_e$ and ${}^b\boldsymbol{\omega}_e$ are respectively, the translational and rotational end-effector velocities in $\mathcal{F}_b$. ${}^w\boldsymbol{\upsilon}_b$ and ${}^w\boldsymbol{\omega}_b$ denote respectively, the translational and rotational velocities of the base frame $\mathcal{F}_b$ with respect to the inertial frame $\mathcal{F}_w$, and ${}^b\mathbf{t}_{re}$ is the position vector of the end-effector $\boldsymbol{r}_e$ in the frame $\mathcal{F}_b$.

Equation (4.3) can be rewritten as follows

$$ \begin{bmatrix} {}^w\boldsymbol{\upsilon}_e \\ {}^w\boldsymbol{\omega}_e \end{bmatrix} = \begin{bmatrix} {}^w\mathbf{R}_b & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^w\mathbf{R}_b \end{bmatrix} \begin{bmatrix} {}^b\boldsymbol{\upsilon}_e \\ {}^b\boldsymbol{\omega}_e \end{bmatrix} + \begin{bmatrix} \mathbf{I}_3 & -\left[{}^w\mathbf{R}_b\,{}^b\mathbf{t}_{re}\right]_\times \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} {}^w\boldsymbol{\upsilon}_b \\ {}^w\boldsymbol{\omega}_b \end{bmatrix} \tag{4.4} $$

where the notation $[.]_\times$ represents a skew symmetric matrix. Using the forward instantaneous kinematics, we have

$$ \begin{bmatrix} {}^b\boldsymbol{\upsilon}_e \\ {}^b\boldsymbol{\omega}_e \end{bmatrix} = {}^b\mathbf{J}_m\dot{\mathbf{q}}_m \qquad \text{and} \qquad \begin{bmatrix} {}^w\boldsymbol{\upsilon}_b \\ {}^w\boldsymbol{\omega}_b \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{T}(\phi_b) \end{bmatrix} \begin{bmatrix} {}^w\dot{\boldsymbol{t}}_b \\ \dot{\boldsymbol{\phi}}_b \end{bmatrix} \tag{4.5} $$

where ${}^{b}\mathbf{J}_{m}$ is the Jacobian of the considered end-effector with respect to the base, $\dot{\mathbf{q}}_b = \begin{bmatrix} {}^{w}\dot{\boldsymbol{t}}_b & \dot{\boldsymbol{\phi}}_b \end{bmatrix}^T$ with ${}^{w}\dot{\boldsymbol{t}}_b = [\dot{x}_b\ \dot{y}_b\ \dot{z}_b]^T$ and $\dot{\boldsymbol{\phi}}_b = [\dot{\alpha}_b\ \dot{\beta}_b\ \dot{\gamma}_b]^T$. $\mathbf{T}(\phi_b)$ is the matrix that maps angular velocities to time derivative of the chosen Euler angles. Hence, substituting (4.5) in (4.4) gives the Jacobian of the end-effector in the inertial frame as

$$ {}^{w}\mathbf{J}_{mb} = \begin{bmatrix} \begin{bmatrix} {}^{w}\mathbf{R}_b & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^{w}\mathbf{R}_b \end{bmatrix} {}^{b}\mathbf{J}_{m} & \begin{bmatrix} \mathbf{I}_3 & -\left[{}^{w}\mathbf{R}_b\, {}^{b}\mathbf{t}_{re}\right]_{\times} \mathbf{T}(\phi_b) \\ \mathbf{0}_3 & \mathbf{T}(\phi_b) \end{bmatrix} \end{bmatrix} \tag{4.6} $$

The obtained Jacobian (4.6) establishes, with respect to an inertial frame, a mapping between joints motions of the "serial manipulator" and the base, and global motions of the considered end-effector.

For the base, however, we are more interested in the global rotational motion ${}^{w}\boldsymbol{\omega}_b = [\omega_{bx}\omega_{by}\omega_{bz}]^T$ instead of its local rotational motion $\dot{\boldsymbol{\phi}} = [\dot{\alpha}_b\ \dot{\beta}_b\ \dot{\gamma}_b]^T$. Hence, the Jacobian can be reduced as follows

$$ {}^{w}\mathbf{J}_{mb_{gr}} = \begin{bmatrix} \begin{bmatrix} {}^{w}\mathbf{R}_b & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^{w}\mathbf{R}_b \end{bmatrix} {}^{b}\mathbf{J}_{m} & \begin{bmatrix} \mathbf{I}_3 & -\left[{}^{w}\mathbf{R}_b\, {}^{b}\mathbf{t}_{re}\right]_{\times} \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \end{bmatrix} \tag{4.7} $$

with the associated joints velocities vector $\dot{\mathbf{q}}_{mb_{gr}} = \left[\dot{\mathbf{q}}_m^T \left[{}^{w}\boldsymbol{\upsilon}_b^T\ {}^{w}\boldsymbol{\omega}_b^T\right]\right]^T$, where the sub-subscript "$_{gr}$" stands for global rotational motion. What remains now is to relate the base's motions to the legs motions.

4.1.2.2 Humanoid's Lower Body Kinematics

Given that a biped humanoid ensures its locomotion by relying on contacts between feet and the ground [134, 16], all possible configurations of the base, relative to the support foot, will be functions of the stance leg joints configuration, as shown in Figure4.2b. Thus, to determine the joints velocity of the base $\dot{\mathbf{q}}_b$ as functions of the stance leg joints, let us write the velocity screw of leg's end-effector with respect to the world frame $\mathcal{F}_w$ as follows

$$ \begin{bmatrix} {}^{w}\boldsymbol{\upsilon}_l \\ {}^{w}\boldsymbol{\omega}_l \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & -\left[{}^{w}\mathbf{R}_b\, {}^{b}\mathbf{t}_{lg}\right]_{\times} \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} {}^{w}\boldsymbol{\upsilon}_b \\ {}^{w}\boldsymbol{\omega}_b \end{bmatrix} + \begin{bmatrix} {}^{w}\mathbf{R}_b & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^{w}\mathbf{R}_b \end{bmatrix} {}^{b}\mathbf{J}_{lg}\dot{\mathbf{q}}_{lg} \tag{4.8} $$

where ${}^{b}\mathbf{J}_{lg}$ is the Jacobian of stance leg's end-effector expressed in the base frame $\mathcal{F}_b$. $\dot{\mathbf{q}}_{lg}$ is the leg's joints velocities vector and ${}^{b}\mathbf{t}_{lg}$ is the position vector of the leg's end-effector with respect to $\mathcal{F}_b$.

Applying Equation (4.8) to the stance foot while assuming no slippage on the ground, and using (4.5), it can be shown that $\dot{\mathbf{q}}_b$ as a function of $\dot{\mathbf{q}}_{lg}$ will be given by

$$ \dot{\mathbf{q}}_b = -\begin{bmatrix} {}^{w}\mathbf{R}_b & \left[{}^{w}\mathbf{R}_b\, {}^{b}\mathbf{t}_{lg}\right]_{\times} {}^{w}\mathbf{R}_b \\ \mathbf{0}_3 & \mathbf{T}(\phi)^{-1\,w}\mathbf{R}_b \end{bmatrix} {}^{b}\mathbf{J}_{lg}\dot{\mathbf{q}}_{lg} \tag{4.9} $$

The motion of the base as function of the leg is also given by

$$\begin{bmatrix} {}^{w}\boldsymbol{v}_b \\ {}^{w}\boldsymbol{\omega}_b \end{bmatrix} = \underbrace{-\begin{bmatrix} {}^{w}\mathbf{R}_b & \left[{}^{w}\mathbf{R}_b\, {}^{b}\mathbf{t}_{lg}\right]_{\times} {}^{w}\mathbf{R}_b \\ \mathbf{0}_3 & {}^{w}\mathbf{R}_b \end{bmatrix}}_{{}^{w}\mathbf{J}_{lg}} {}^{b}\mathbf{J}_{lg}\dot{\mathbf{q}}_{lg} \tag{4.10}$$

where ${}^{w}\mathbf{J}_{lg}$ represents the Jacobian mapping the stance leg's joints velocities to the base velocity with respect to the inertial frame $\mathcal{F}_w$.

4.2 Visual Servoing based Humanoid's Walking

Let us consider now a visually controlled walking application, where using its embedded camera, a humanoid robot autonomously walks from a given position towards a desired position with respect to a target object. Achieving this task requires a coordinated control between the lower body and upper body through the support leg's joints and the neck's joints, respectively.

Our first approach, illustrated in Figure 4.3, consists of computing the neck's joints and base velocities using the Jacobian (4.6) or (4.7), then the obtained neck's joints velocities are sent directly as set-points to the robot low-level controller. The computed base's velocities, on the other hand, are the references to be tracked by the locomotion module, which will generate the legs joints motion after accounting first for the balance and stability, and other kinematic constraints.

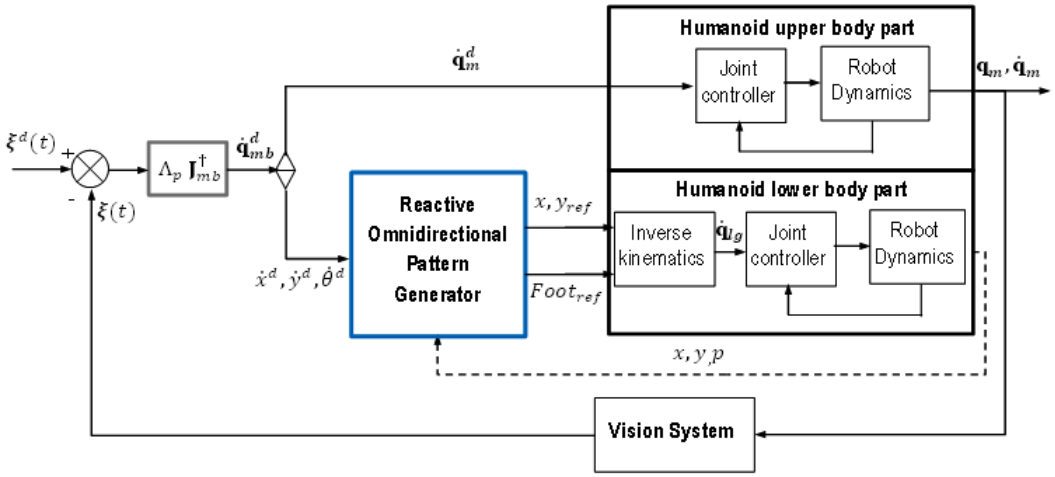


Figure 4.3: Block diagram of a possible visual servoing scheme on humanoid robot

Let $\boldsymbol{e}(t) = \boldsymbol{\xi}(t) - \boldsymbol{\xi}^d$ be visual task function representing the considered robotic task, where $\boldsymbol{\xi}(t)$ and $\boldsymbol{\xi}^d$ denote the current and desired image features vector, respectively. Using Equations (2.13) and (4.7), the associated visual task Jacobian will be given by

$$\mathbf{J}_{e_{mb}} = \mathbf{L}_{\xi_{tsk}}\, {}^{c}\mathbf{W}_{w}\, {}^{w}\mathbf{J}_{mb} \tag{4.11}$$

where $\mathbf{L}_{\xi_{tsk}}$ is the interaction matrix associated with the chosen features vector $\boldsymbol{\xi}$. ${}^{c}\mathbf{W}_{w}$ is the twist transformation matrix from $\mathcal{F}_w$ to the camera frame and is given by

$$ {}^{c}\mathbf{W}_{w} = \begin{bmatrix} {}^{c}\mathbf{R}_{w} & [{}^{c}\boldsymbol{t}_{w}]_{\times}\, {}^{c}\mathbf{R}_{w} \\ \mathbf{0}_{3} & {}^{c}\mathbf{R}_{w} \end{bmatrix} \tag{4.12} $$

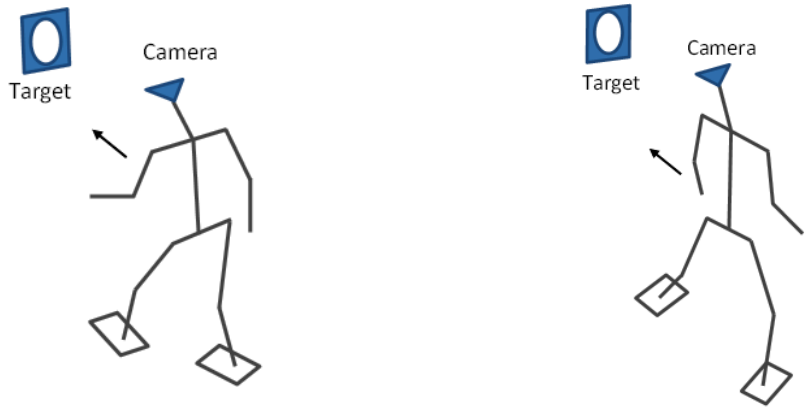
with ${}^{c}\mathbf{R}_{w}$ and ${}^{c}\boldsymbol{t}_{w}$ respectively, the rotation and translation from $\mathcal{F}_w$ to the camera frame, expressed in the camera frame. In this case $\mathcal{F}_w$ is taken to be the stance foot frame. ${}^{w}\mathbf{J}_{mb}$ is given by (4.6) within which ${}^{b}\mathbf{J}_{m}$ corresponds to the Jacobian of the head's chain ${}^{b}\mathbf{J}_{head}$, with $\mathbf{q}_m = \mathbf{q}_{neck} = [\theta_{1n}\ \theta_{2n}]^T$ and where θ_{in} are the neck joints (pan and tilt).

Using Equations (2.14) and (4.11), the joints motions to perform the considered task can be computed as follows

$$ \dot{\mathbf{q}}_{mb} = \mathbf{J}^{\dagger}_{e_{mb}} \dot{\boldsymbol{e}}(t) \tag{4.13} $$

where $\dot{\mathbf{q}}_{mb} = [\dot{\theta}_{1n}\ \dot{\theta}_{2n}\ \dot{x}_b\ \dot{y}_b\ \dot{z}_b\ \dot{\alpha}_b\ \dot{\beta}_b\ \dot{\gamma}_b]^T$ or $\dot{\mathbf{q}}_{mb_{gr}} = \left[\dot{\theta}_{1n}\, \dot{\theta}_{2n}\, \dot{x}_b\, \dot{y}_b\, \dot{z}_b\, \omega_{bx}\, \omega_{by}\, \omega_{bz}\right]^T$ if $\mathbf{J}_{e_{mb_{gr}}}$ were used, and $\mathbf{J}^{\dagger}_{e_{mb}}$ is an appropriate pseudo-inverse of the task Jacobian $\mathbf{J}_{e_{mb}}$.

Note that only six DoFs are required to uniquely define the pose of an object in the manifold $SE(3)$. Given that the humanoid's embedded camera has a number of DoFs (here eight) greater than those required for the task, the robot will be, therefore, redundant with respect to the considered task [127]. Hence, there are multiple ways for the humanoid to approach the target object, provided that its camera is pointing in the right direction. This is illustrated in Figure 4.4a, where the robot is walking with its torso facing the target and in Figure 4.4b, where the robot is approaching sideways the target.



(a) Approaching the target while facing it

(b) Approaching the target sideways

Figure 4.4: Illustration of two different ways of approaching the target by a humanoid robot

Thus, in order to guarantee an efficient humanoid's approach towards the target, this redundancy problem has to be addressed. Another problem to be considered is how to integrate the walking motion. Indeed, the locomotion module, for balance and stability sake, will constrain the motion of the base; and due to the unavoidable sway motion introduced, the reference velocity could only be tracked on average [19], [23]. How this will affect the task execution and how it can possibly be compensated have also to be considered.

4.2.1 Redundancy Resolution of Visual Task

Depending on the dimension of the configuration space with respect to either the number of the end-effector's DoFs or the dimension of a given task, the notion of redundancy can be defined differently [29], [146], [147], [35]. Particularly, we will be more concerned by the definition referring to the dimension of the task. As a consequence of the robot's redundancy, the inverse kinematics problem will admit an infinite number of solutions satisfying the desired task [127].

The common approach to resolve this problem consists of exploiting the null-space of the main task in order to perform additional tasks or to achieve auxiliary objective, such as optimizing a given criterion [148]. For instance, the main task can be positioning the camera or tracking a given trajectory, while the auxiliary task(s) could be improving the manipulability [149], avoiding joints limits [150, 151, 114, 152], avoiding occlusions or/and obstacles [153, 154].

Thus, the general solution to Equation (4.13) will be given by [148]

$$\dot{\mathbf{q}}_{mb} = {}^{w}\mathbf{J}^{\dagger}_{mb}\dot{\boldsymbol{r}}_{e} + \mathbf{P}_{mb} \cdot \boldsymbol{g}(q) \tag{4.14}$$

where $\mathbf{P}_{mb} = (\mathbf{I}_n - {}^{w}\mathbf{J}^{\dagger}_{mb}{}^{w}\mathbf{J}_{mb})$ is an orthogonal projection operator onto the null-space of the main task (see, for instance, ref. [154] and [152] for other projectors). And $\boldsymbol{g}(q)$ is the joints velocity vector resulting from the secondary task. The actions of $\boldsymbol{g}(q)$, while orthogonally projected, will not affect $\dot{\boldsymbol{r}}_e$ given that [126] ${}^{w}\mathbf{J}_{mb}\dot{\mathbf{q}}_{mb} = {}^{w}\mathbf{J}_{mb}({}^{w}\mathbf{J}^{\dagger}_{mb}\dot{\boldsymbol{r}}_{e} + \mathbf{P} \cdot \boldsymbol{g}(q)) = \dot{\boldsymbol{r}}_{e}$; therefore, it will merely result in an internal motion.

As additional tasks exploiting the redundancy, we will consider the alignment between the direction of sight and the humanoid's body, the avoidance of joints limits and the manipulability [149] for the grasping task. The internal motion required to achieve these objectives will result from optimizing configuration-dependent cost functions, using the gradient projective method (GPM).

4.2.1.1 Gradient Projective Method

Let $h\,(q(t))$ be a cost function to be minimized and whose time derivative is given by

$$\dot{h}\,(q) = \nabla^{T}_{q} h\,(q)\,\dot{\mathbf{q}} \tag{4.15}$$

where $\nabla^{T}_{q} h\,(q)$ is the gradient of $h\,(q)$. Substituting (4.14) in (4.15) for $\dot{\mathbf{q}}(t)$ gives

$$\dot{h}(q) = \nabla_q^T h(q)\, {}^w\mathbf{J}_{mb}^{\dagger}\dot{\boldsymbol{r}}_e + \nabla_q^T h(q)\, \mathbf{P}_{mb} \cdot \boldsymbol{g}(q) \tag{4.16}$$

with the first term in the right hand side of (4.16) representing the variation of $h(q)$ due to the main task, while the second term is the variation of $h(q)$ due to the internal motion. Provided that the main task is satisfied, a maximum decrease of $h(q)$ can be obtained by choosing $\boldsymbol{g}(q)$ as follows [52]

$$\boldsymbol{g}(q) = -k\left(\nabla_q^T h(q)\, \mathbf{P}_{mb}\right)^T \tag{4.17}$$

where k is a positive gain.
Hence, substituting (4.17) in (4.16) gives $\dot{h}(q) = -k\left(\nabla_q^T h(q)\, \mathbf{P}_{mb} \cdot \mathbf{P}_{mb}^T \nabla_q h(q)\right) \leq 0$ for the internal motion [127], which implies a continuous minimization of $h(q)$. Finally, substituting (4.17) in (4.14) and recalling that $\mathbf{P}_{mb}$ is symmetric and idempotent [155], the velocity commands, while accounting for secondary objective, will now be given by

$$\dot{\mathbf{q}}_{mb} = {}^w\mathbf{J}_{mb}^{\dagger}\dot{\boldsymbol{r}}_e - k \cdot \mathbf{P}_{mb} \cdot (\nabla_q h(q)) \tag{4.18}$$

In the case where several objective functions $h_i(q)$ are considered, they can be linearly combined to yield a global function as follows

$$h(q) = \sum \alpha_i h_i(q) \tag{4.19}$$

where α_i denotes the weighting factor of individual cost function in the global function. Thus, the global $g(t)$ will be

$$\boldsymbol{g}(t) = \sum_i \alpha_i g_i(t) = \sum_i \alpha_i \left(\nabla_q h_i(q)\right) \tag{4.20}$$

After presenting the GPM and showing how it will be applied to multiple objectives; in the sequel, we show how the auxiliary tasks considered here will be implemented.

4.2.1.2 Auxiliary Tasks

Humanoid Sight's Direction and Body Alignment

We have shown previously (Figure 4.4) that the humanoid robot, having an omnidirectional locomotion, could approach the target with different orientations of its base as long as its camera points in the object's direction. Aligning the humanoid's sight direction with its body aims at solving that problem. Thus, it will allow the robot, for instance, to walk to or track a target while autonomously facing it.

In that respect, let us assume an upright posture of the robot and let us consider the camera's orientation about the vertical axis. Its variation is subject to changes either of the neck's joint θ_{1n} or of the rotation angle γ_b of the locomotion module. Hence, for any value of θ_{1n} there is a corresponding value of γ_b satisfying that configuration. Therefore, there exist an infinite number of solutions to the inverse kinematics.

Using the method described previously, minimizing the difference between θ_{1n} and γ_b ($min\ |\theta_{1n} - \gamma_b|$) may result into an alignment of the body and the sight direction. Thus, the following cost function is considered

$$h_{sba}(q) = \frac{1}{2}(\theta_{1n} - \gamma_b)^2 \tag{4.21}$$

where the subscript "*sba*" stands for sight-body alignment. $h_{sba}(q)$ is minimal when $\theta_{1n} = \gamma_b$ and its gradient $\boldsymbol{g}_{sba}(q)$ can be obtained as follows

$$\begin{cases} \frac{\partial h_{sba}}{\partial q_1} & = g_{sba_1} = (\theta_{1n} - \gamma_b) \\ \vdots & \\ \frac{\partial h_{sba}}{\partial q_i} & = g_{sba_i} = 0 \qquad \text{for } i = 2, \ldots, 7 \\ \vdots & \\ \frac{\partial h_{sba}}{\partial q_8} & = g_{sba_8} = -(\theta_{1n} - \gamma_b) \end{cases} \tag{4.22}$$

and

$$\nabla_q h_{sba}(q) = \boldsymbol{g}_{sba}(q) = \begin{bmatrix} g_{sba_1} & \cdots & g_{sba_i} & \cdots & g_{sba_8} \end{bmatrix}^T \tag{4.23}$$

with $i = 2, \ldots, 7$.

Using Equations (4.18) and (4.23), the control law achieving an exponential decrease of the visual task $\boldsymbol{e}(t)$, while including the "body and sight alignment" task can now be written as

$$\dot{\mathbf{q}}_{mb} = -\widehat{\mathbf{J}}^{\dagger}_{e_{mb}} \Lambda_p \boldsymbol{e}(t) - \alpha_{sba} \mathbf{P}_{mb} \cdot \boldsymbol{g}_{sba}(q) \tag{4.24}$$

where $\dot{\mathbf{q}}_{mb} = [\dot{\theta}_{1n}\ \dot{\theta}_{2n}\ \dot{x}_b\ \dot{y}_b\ \dot{z}_b\ \dot{\alpha}_b\ \dot{\beta}_b\ \dot{\gamma}_b]^T$, Λ_p is a matrix of proportional gains associated with the decrease of $\boldsymbol{e}(t)$, such that $\dot{\boldsymbol{e}}(t) = -\Lambda_p \boldsymbol{e}(t)$, and $\alpha_{sba} = \alpha_{sba}(\theta_{1n}, \gamma_b)$ is an adaptive gain designed to reduce the effect of the gradient when close to the desired alignment, it is given by

$$\alpha_{sba} = \eta \left(1 - exp(-\frac{(\theta_{1n} - \gamma_b)^2}{\triangle q_{1\gamma}^2}) \right) \tag{4.25}$$

with η a constant and $\triangle q_{1\gamma}$ the admissible difference between θ_{1n} and γ_b.

Joint Limits Avoidance

The joint limits are physical limits restraining the extension of the operational space of the robot [156]. If the motion computed by the visual controller exceeds the joints limits, the servoing process may fail, since no motion can be performed beyond the physical limits of the joints. Thus, an additional task aiming at constraining the joint's values within their physical limits is considered.

Let q_i^{min} and q_i^{max} denote, respectively, the lower and upper limit associated with the ith joint q_i. The joint limits avoidance scheme should be active only when at least one joint is close to its limits. And it should stay inactive when the joint's values are acceptable, that is when the joint's

values $q_i \in [q_{l_{0^i}}^{min},\ q_{l_{0^i}}^{max}]$ with $q_{l_{0^i}}^{min}$ and $q_{l_{0^i}}^{max}$ the activation thresholds defined as [156, 157, 158]

$$\begin{aligned} q_{l_{0^i}}^{min} &\triangleq q_i^{min} + \rho\Delta q_i \\ q_{l_{0^i}}^{max} &\triangleq q_i^{max} - \rho\Delta q_i \end{aligned} \tag{4.26}$$

where $\Delta_i q_i = q_i^{max} - q_i^{min}$ and ρ is a tuning parameter, within the interval $[0,\ \frac{1}{2}]$ (generally, $\rho = 0.1$). The cost function can thus be defined as

$$h_{jla}(q) = \frac{1}{2}\sum_{i=1}^{n}\frac{\Delta_i^2}{\Delta q_i} \tag{4.27}$$

where the subscript "jla" stands for joint limits avoidance and where

$$\Delta_i = \begin{cases} q_i - q_{l_{0^i}}^{min}, & \text{if } q_i < q_{l_{0^i}}^{min} \\ q_i - q_{l_{0^i}}^{max}, & \text{if } q_{l_{0^i}}^{max} < q_i \\ 0 & \text{else} \end{cases} \tag{4.28}$$

The cost function's gradient g_{jla} is obtained as follows

$$g_{jla_i}(q) = \frac{\partial h_{jla}(q)}{\partial q_i} = \frac{\Delta_i}{\Delta q_i} \tag{4.29}$$

$$\mathbf{g}_{jla} = [\ g_{jla_1} \quad \cdots \quad g_{jla_n}\ ]^T \tag{4.30}$$

Finally, the resulting velocity commands will be obtained using (4.18).

Manipulability

The manipulability concept was proposed by *Yoshikawa* [149]. This concept provides information on how far a robot is from a singular configuration. Therefore, it has been used in redundancy resolution frameworks to avoid singular configurations.

Similarly, we will improve the robot's dexterity, for instance, for the grasping task using the following manipulabilty measure [149]

$$\omega_r = \sqrt{det\left({\mathbf{J}^T}_{mb}(\mathbf{q})\mathbf{J}_{mb}(\mathbf{q})\right)} \tag{4.31}$$

which represents the volume of the ellipsoid formed by the set of operational velocities $\dot{\boldsymbol{r}}_e$ realizable by the robot's arm at a given configuration $\mathbf{q}$ when the generalized velocities $\dot{\mathbf{q}}_{mb}$ are bounded in a unit ball $\|\dot{\mathbf{q}}_{mb}\| \leq 1$. The ellipsoid axes are given by the singular values $\sigma_1,\ \sigma_2, \ldots,\ \sigma_{n_{mb}}$ of the Jacobian $\mathbf{J}_{mb}(\mathbf{q})$ that maps the generalized velocities to the operational velocities [127].

Requiring high manipulability can ensure to the robot's arm the ability to easily move in ar-

bitrary directions. For that purpose, we defined the following cost function to be minimized

$$h_{man}(q) = \frac{1}{\omega_r} = \frac{1}{\sqrt{det\left(\mathbf{J}^T{}_{mb}(q)\mathbf{J}_{mb}(q)\right)}} \tag{4.32}$$

Dropping the dependency for simplicity in writing, the gradient g_{man} is obtained as

$$g_{man_i} = \frac{\partial h_{man}}{\partial q_i} = -\frac{1}{2}\left[det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right]^{-\frac{3}{2}} \cdot \frac{\partial\left[det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right]}{\partial q_i} \tag{4.33}$$

Note that the gradient obtained from (4.33) will be infinite at singular configurations where $det\left(\mathbf{J}_{mb}\right) = 0$. Hence, to avoid a numerical division by zero, it is defined a saturated determinant as follows [150]

$$det_{sat}\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right) = \begin{cases} det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right) & \text{if } \left|det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right| \geq \varepsilon \\ \varepsilon \cdot sgn\left(det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right) & \text{if } \left|det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right| < \varepsilon \end{cases} \tag{4.34}$$

where ε is a arbitrarily small constant and sgn denotes the signum function. Thus, the gradient of the chosen manipulability cost function will be computed as

$$\mathbf{g}_{man} = -\frac{1}{2}\left[det_{sat}\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right]^{-\frac{3}{2}} \cdot \nabla_q\left[det\left(\mathbf{J}^T{}_{mb}\mathbf{J}_{mb}\right)\right] \tag{4.35}$$

Finally, the resulting velocity commands will be obtained using (4.18).

4.2.2 Integration of Walking Motion

Until now, the walking motion has been implicitly described through the motion of the base, namely ($\dot{x}_b$, $\dot{y}_b$, $\dot{z}_b$, $\dot{\alpha}_b$, $\dot{\beta}_b$, $\dot{\gamma}_b$). Using this model, it was possible to compute the required motion to be sent as control commands to the base in order to perform a given task. However, the computed motion describes nothing but the motion of a free body in space and therefore, it does not reflect any walking constraint.

Indeed, during the task execution, the base's motion results from the locomotion part, which is known to have constrained motions. First, the gravity pushes the humanoid against the ground; from contacts between the latter and the humanoid's feet result interactions forces without which the bipedal locomotion cannot take place. This implies constrained z displacements of the base within a range, given that the stance foot which has to be in contact with the ground has a limited length. Furthermore, as mentioned in Section 3.2.1, the ZMP (for dynamic stability) is constrained to lie within the support polygon determined by the feet [17, 159]; this imposes constraints on the CoM trajectories along the X and Y axes. Moreover, the pattern generator used to generate the CoM trajectories relies on a simplified dynamic model of the robot (3D LIPM or Cart-Table model [160], [113]) which does not account explicitly for angular momenta due to the rotational motions $\dot{\alpha}_b$ and $\dot{\beta}_b$ of the base, these motions have thus to be constrained

at a minimum. Consequently, as presented in Section 3.3.3, the gait generator will only take translations along X and Y, ($\dot{x}_b$ and $\dot{y}_b$), and rotation about Z ($\dot{\gamma}_b$) as reference inputs. Thus, the motion due to the roll angle ($\dot{\alpha}_b$), the pitch angle ($\dot{\beta}_b$) and the translation $\dot{z}_b$ will be considered as disturbance to the task. It, therefore, needs to be compensated.

Consider first, at the kinematic level, a given velocity ${}^{w}\dot{\boldsymbol{r}}_{e_d}$ to be performed by any upper body end-effector, the required upper body joints and base velocities can be obtained using (4.14), which is recalled here. Thus we have

$$\dot{\mathbf{q}}_{mb} = {}^{w}\mathbf{J}_{mb}^{\dagger}\,{}^{w}\dot{\boldsymbol{r}}_{e_d} + \mathbf{P}_{mb}\cdot\mathbf{g}(q) \tag{4.36}$$

where the control velocities $\dot{\mathbf{q}}_{mb} = [\ \dot{\mathbf{q}}_m^T \quad \dot{\mathbf{q}}_b^T\ ]^T$, based on the considerations above, can be rewritten as

$$\dot{\mathbf{q}}_{mb} = \left[\ \dot{\mathbf{q}}_m^T \quad (\dot{\mathbf{q}}_{b\_walk} + \dot{\mathbf{q}}_{b\_dist})^T\ \right]^T \tag{4.37}$$

with $\dot{\mathbf{q}}_b = \dot{\mathbf{q}}_{b\_walk} + \dot{\mathbf{q}}_{b\_dist}$, where $\dot{\mathbf{q}}_{b\_walk}$, respectively, $\dot{\mathbf{q}}_{b\_dist}$ represent the controlled motion of the base and the disturbance (the undesired motion), as seen from the gait generator perspective. They are given by

$$\begin{cases} \dot{\mathbf{q}}_{b\_walk} &= [\ \dot{x}_b \quad \dot{y}_b \quad 0 \quad 0 \quad 0 \quad \dot{\gamma}_b\ ]^T \\ \dot{\mathbf{q}}_{b\_dist} &= [\ 0 \quad 0 \quad \dot{z}_b \quad \dot{\alpha}_b \quad \dot{\beta}_b \quad 0\ ]^T \end{cases} \tag{4.38}$$

Let us assume that the gait generator can perfectly track its reference velocities ($\dot{x}_b$, $\dot{y}_b$, $\dot{\gamma}_b$), likewise for the considered upper body "manipulator" with $\dot{\mathbf{q}}_m$. The motion ${}^{w}\dot{\boldsymbol{r}}_{e_{ex}}$ executed by the end-effector will be

$${}^{w}\dot{\boldsymbol{r}}_{e_{ex}} = {}^{w}\mathbf{J}_m\dot{\mathbf{q}}_m + {}^{w}\mathbf{J}_b\dot{\mathbf{q}}_{b\_walk} \tag{4.39}$$

where ${}^{w}\mathbf{J}_m$ and ${}^{w}\mathbf{J}_b$ are Jacobian matrices formed by the columns of ${}^{w}\mathbf{J}_{mb}$ associated with $\dot{\mathbf{q}}_m$ and $\dot{\mathbf{q}}_b$, respectively. Since the motion ($\dot{z}_b$, $\dot{\alpha}_b$, $\dot{\beta}_b$) has been ignored, it results an error between ${}^{w}\dot{\boldsymbol{r}}_{e_d}$ and ${}^{w}\dot{\boldsymbol{r}}_{e_{ex}}$ given by

$${}^{w}\dot{\boldsymbol{r}}_{e_d} - {}^{w}\dot{\boldsymbol{r}}_{e_{ex}} = {}^{w}\mathbf{J}_b\dot{\mathbf{q}}_{b\_dist} \tag{4.40}$$

Given that $\dot{\mathbf{q}}_{b\_walk}$ is constrained, as stated above, in order to execute the desired end-effector velocity ${}^{w}\dot{\boldsymbol{r}}_{e_d}$, we can only compensate for the error by recomputing the upper body reference velocities. Hence, we can write

$${}^{w}\dot{\boldsymbol{r}}_{e_d} = {}^{w}\mathbf{J}_m\dot{\mathbf{q}}_{m\_walk} + {}^{w}\mathbf{J}_b\dot{\mathbf{q}}_{b\_walk} \tag{4.41}$$

where ${}^{w}\mathbf{J}_m\dot{\mathbf{q}}_{m\_walk} = {}^{w}\mathbf{J}_m\dot{\mathbf{q}}_m + {}^{w}\mathbf{J}_b\dot{\mathbf{q}}_{b\_dist}$ represents, in a general sense, the required motion contribution from the upper body to a desired end-effector velocity while the humanoid is walking. Now the upper body joint velocities after compensation can be computed as

$$\dot{\mathbf{q}}_{m\_walk} = {}^{w}\mathbf{J}_m^{\dagger}\left({}^{w}\dot{\boldsymbol{r}}_{e_d} - {}^{w}\mathbf{J}_b\dot{\mathbf{q}}_{b\_walk}\right) + \mathbf{P}_m\cdot\mathbf{g}_m(q) \tag{4.42}$$

where ${}^{w}\mathbf{J}_{m}^{\dagger}$ is the pseudo-inverse of ${}^{w}\mathbf{J}_{m}$, $\mathbf{P}_{m}$ is its associated projection operator and $\mathbf{g}_{m}(q)$ is the joint velocities resulting from an auxiliary task.

Note that $\dot{\mathbf{q}}_{m\_walk}$ is the control upper body joints velocity subject to the execution of the walking motion. This means that regardless the tracking performances of the gait generator (assumed to be ideal before), using the actual base velocity $\dot{\mathbf{q}}_{b\_walk}$ instead of its reference value, the required upper body velocity to perform a given velocity can be computed using (4.42). This equation can conveniently be rewritten in terms of the actuated leg's as follows

$$\dot{\mathbf{q}}_{m\_walk} = {}^{w}\mathbf{J}_{m}^{\dagger}\left({}^{w}\dot{\boldsymbol{r}}_{e_d} - {}^{w}\mathbf{J}_{re\_lg}\cdot\dot{\mathbf{q}}_{lg\_walk}\right) + \mathbf{P}_{m}\cdot\mathbf{g}_{m}(q) \tag{4.43}$$

where $\dot{\mathbf{q}}_{lg\_walk}$ is the joints velocities vector of the support leg with embedded balance and stability constraints, since it results from the gait generator. In practice, $\dot{\mathbf{q}}_{lg\_walk}$ can be directly estimated from the joints measurements. ${}^{w}\mathbf{J}_{re\_lg}$ is the support leg's Jacobian related to $\dot{\boldsymbol{r}}_{e}$ and expressed in $\mathcal{F}_{w}$. From (4.6) and (4.9), it can be obtained as follows

$${}^{w}\mathbf{J}_{re\_lg} = -\begin{bmatrix} \mathbf{I}_3 & -\left[{}^{w}\mathbf{R}_{b}{}^{b}\mathbf{t}_{re}\right]_{\times}\mathbf{T}(\phi) \\ \mathbf{0}_3 & \mathbf{T}(\phi) \end{bmatrix}\begin{bmatrix} {}^{w}\mathbf{R}_{b} & \left[{}^{w}\mathbf{R}_{b}{}^{b}\mathbf{t}_{lg}\right]_{\times}{}^{w}\mathbf{R}_{b} \\ \mathbf{0}_3 & \mathbf{T}(\phi)^{-1w}\mathbf{R}_{b} \end{bmatrix}{}^{b}\mathbf{J}_{lg}$$

which finally gives

$${}^{w}\mathbf{J}_{re\_lg} = -\begin{bmatrix} {}^{w}\mathbf{R}_{b} & -\left[{}^{w}\mathbf{R}_{b}({}^{b}\mathbf{t}_{re} - {}^{b}\mathbf{t}_{lg})\right]_{\times}{}^{w}\mathbf{R}_{b} \\ \mathbf{0}_3 & {}^{w}\mathbf{R}_{b} \end{bmatrix}{}^{b}\mathbf{J}_{lg} \tag{4.44}$$

Thus, the Jacobian ${}^{w}\mathbf{J}_{re\_lg}$ will map the motion of any support leg's joint to the corresponding velocity of the considered upper body end-effector.

Now that the redundancy has been handled and the walking motion integrated, we can now apply this control scheme to perform some tasks.

4.3 Visual Servoing based Humanoid's Positioning

Let us consider now a positioning task, where using its embedded camera, a humanoid robot autonomously walks from a given position towards a desired position relative to a target object and stop at that desired pose.

Based on the above formulation, our approach to solve this problem, as illustrated in Figure 4.5, consists of computing first the neck's and base's joints velocities using the Jacobian (4.6), then the obtained base's velocities are sent as reference commands to be tracked by the locomotion module. The latter will thus generate the legs joints motion after accounting for the balance and gait stability, and other kinematic constraints. The neck's joints velocities, on the other hand, are dropped and new values are recomputed using (4.43) in order to compensate for any error in the task due to the difference between the reference base's motion and the actual one,

which is subject to various constraints.

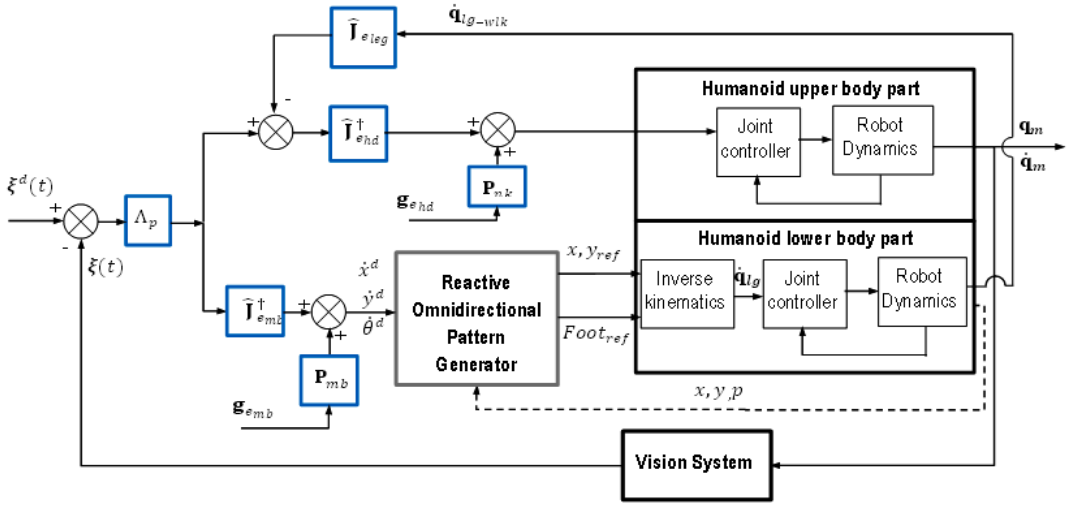


Figure 4.5: Visual servoing scheme for positioning task on humanoid robot

4.3.1 Modeling

Let us define $\boldsymbol{e}(t) \triangleq \boldsymbol{\xi}(t) - \boldsymbol{\xi}^d \in \mathbb{R}^k$ as the visual task function representing the positioning task, where $\boldsymbol{\xi}(t)$ and $\boldsymbol{\xi}^d$ keep the same meaning as before. Using (2.13), the time derivative of the task function can be written as

$$\dot{\boldsymbol{e}}(t) = \mathbf{J}_{e_{mb}} \dot{\mathbf{q}}_{mb} + \frac{\partial \boldsymbol{e}(t)}{\partial t} \tag{4.45}$$

where $\mathbf{J}_{e_{mb}} \in \mathbb{R}^{k \times n_{mb}}$ is the visual task Jacobian associated with $\boldsymbol{e}(t)$. Similarly to (4.11), from Equations (2.13) and (4.6), $\mathbf{J}_{e_{mb}}$ will be given by

$$\mathbf{J}_{e_{mb}} = \mathbf{L}_{\xi_{tsk}} {}^{c}\mathbf{W}_{w} {}^{w}\mathbf{J}_{mb} \tag{4.46}$$

The Jacobian ${}^{w}\mathbf{J}_{mb}$ has the form of (4.6) and it is given by

$$\begin{aligned} {}^{w}\mathbf{J}_{mb} &= \left[\begin{bmatrix} {}^{w}\mathbf{R}_b & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^{w}\mathbf{R}_b \end{bmatrix} {}^{b}\mathbf{J}_{c\_hd} \quad \begin{bmatrix} \mathbf{I}_3 & -\left[{}^{w}\mathbf{R}_b {}^{b}\mathbf{t}_c \right]_{\times} \mathbf{T}(\phi_b) \\ \mathbf{0}_3 & \mathbf{T}(\phi_b) \end{bmatrix} \right] \\ &= \begin{bmatrix} {}^{w}\mathbf{J}_{c\_hd} & {}^{w}\mathbf{J}_{c\_b} \end{bmatrix} \end{aligned} \tag{4.47}$$

where ${}^{b}\mathbf{t}_c \in \mathbb{R}^3$ corresponds to the position vector of the origin of the camera's frame $\mathcal{F}_c$ expressed in the base frame $\mathcal{F}_b$, and ${}^{b}\mathbf{J}_{c\_hd} \in \mathbb{R}^{6 \times n_m}$ is its associated Jacobian along the head's chain. If there is no joint in the torso and we have a pan/tilt head, the vector $\mathbf{q}_m \in \mathbb{R}^{n_m}$ of the n_m joints along the head chain would be $\mathbf{q}_{nk} = [\theta_{1n}\, \theta_{2n}]^T$ with θ_{in} the neck joints.

Assuming that the object is motionless ($\frac{\partial \boldsymbol{e}(t)}{\partial t} = 0$), the time-variation of the task function will

now be related to the robot's joints motion by

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_{\xi_{tsk}}\, {}^{c}\mathbf{W}_{w} \underbrace{\begin{bmatrix} {}^{w}\mathbf{J}_{c\_hd} & {}^{w}\mathbf{J}_{c\_b} \end{bmatrix}}_{{}^{w}\mathbf{J}_{mb}} \begin{bmatrix} \dot{\mathbf{q}}_{nk} \\ \dot{\mathbf{q}}_{b} \end{bmatrix} \tag{4.48}$$

where $\dot{\mathbf{q}}_{mb} = \begin{bmatrix} \dot{\mathbf{q}}_{nk}^T & \dot{\mathbf{q}}_{b}^T \end{bmatrix}^T = [\dot{\theta}_{1n}\, \dot{\theta}_{2n}\, \dot{x}_b\, \dot{y}_b\, \dot{z}_b\, \dot{\alpha}\, \dot{\beta}\, \dot{\gamma}]^T$. In case the Jacobian ${}^{w}\mathbf{J}_{mb_{gr}}$ (4.7) is utilized instead of ${}^{w}\mathbf{J}_{mb}$, the task Jacobian will be $\mathbf{J}_{e_{mb_{gr}}}$ which will yield the velocities vector $\dot{\mathbf{q}}_{mb_{gr}} = \begin{bmatrix} \dot{\theta}_{1n}\, \dot{\theta}_{2n}\, \dot{x}_b\, \dot{y}_b\, \dot{z}_b\, \omega_{bx}\, \omega_{by}\, \omega_{bz} \end{bmatrix}^T$.

4.3.2 Control

Assuming that the robot has low-level controller able to execute velocity commands, using (4.48) and (2.14), the control law computing the required joints motions to perform the given task, while achieving an exponential decrease of the error ($\dot{\boldsymbol{e}}(t) = -\Lambda_p \boldsymbol{e}(t)$) can be written as follows

$$\begin{bmatrix} \dot{\mathbf{q}}_{nk} \\ \dot{\mathbf{q}}_{b} \end{bmatrix} = \dot{\mathbf{q}}_{mb} = -\widehat{\mathbf{J}}^{\dagger}_{e_{mb}} \Lambda_p \boldsymbol{e}(t) + \mathbf{P}_{e_{mb}} \cdot \mathbf{g}_{e_{mb}}(q) \tag{4.49}$$

where $\widehat{\mathbf{J}}^{\dagger}_{e_{mb}}$ denotes an estimate of the pseudo-inverse of $\mathbf{J}_{e_{mb}}$, $\mathbf{P}_{e_{mb}} \in \mathbb{R}^{n_{mb} \times n_{mb}}$ is its associated projection operator, $\mathbf{g}_{e_{mb}}(q) \in \mathbb{R}^{n_{mb}}$ is the joint velocities resulting from an auxiliary task, for instance, using joint limits avoidance to minimize as much as possible the roll, pitch and vertical motions of the base.

The computed base's velocities are the reference commands to be executed by the locomotion module (reactive omnidirectional pattern generator). We recall that the latter uses as inputs the velocities of the center of mass (CoM). Thus, from the base's velocity, the reference velocities of the CoM are computed as

$$\begin{bmatrix} {}^{w}\boldsymbol{\upsilon}_{CoM} \\ {}^{w}\boldsymbol{\omega}_{CoM} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & -\left[{}^{w}\mathbf{R}_b\, {}^{b}\mathbf{t}_{CoM}\right]_{\times} \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} {}^{w}\boldsymbol{\upsilon}_{b} \\ {}^{w}\boldsymbol{\omega}_{b} \end{bmatrix} \tag{4.50}$$

where $\begin{bmatrix} {}^{w}\boldsymbol{\upsilon}_{CoM}^T & {}^{w}\boldsymbol{\omega}_{CoM}^T \end{bmatrix}^T \in \mathbb{R}^6$ is the velocity screw of the frame Σ_{CoM} attached to the center of mass and ${}^{b}\mathbf{t}_{CoM} \in \mathbb{R}^3$ is the position vector of the center of mass with respect to the base frame.

Now, from the obtained center of mass velocities in (4.50), only the components $\dot{x}_{CoM}$, $\dot{y}_{CoM}$ and ω_{CoM_z} are sent to the walking gait generator in order to be tracked. That is:

$$\begin{cases} \dot{C}^{x}_{ref} & = \dot{x}_{CoM} \\ \dot{C}^{y}_{ref} & = \dot{y}_{CoM} \\ \dot{\Theta}_{ref} & = \omega_{CoM\_z} \end{cases} \tag{4.51}$$

The CoM's velocity components $\omega_{CoM\_x}, \omega_{CoM\_y}$ and $\dot{z}_{CoM}$ regarded as disturbance are dropped; this has very little impact on the overall result, since they were indirectly minimized while computing (4.49).

From the CoM and feet trajectories computed by the pattern generator while tracking the reference CoM velocities, the robot's inverse kinematics will produce velocity set-points, denoted $\dot{\mathbf{q}}^*_{lg\_wlk}$, for the legs joints controller. Once executed, the actual legs joints may differ from their set-points due to modeling, control errors and other uncertainties and/or disturbances. Hence, the neck's joints velocities can now be recomputed such that all motions (controlled and uncontrolled) associated with the locomotion part and affecting the task function be compensated.

To that end, let $\dot{\mathbf{q}}_{lg\_wlk}$ be defined as the actual joints velocities vector of the support leg, and let $\dot{\mathbf{q}}_{nk\_wlk}$ be the neck's joint velocities vector compensating for all motion due to the walking part. The time derivative of $\boldsymbol{e}(t)$ can be written as:

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_{\xi_{tsk}}\, {}^{c}\mathbf{W}_{w} \left({}^{w}\mathbf{J}_{c\_hd} \dot{\mathbf{q}}_{nk\_wlk} + {}^{w}\mathbf{J}_{c\_lg} \cdot \dot{\mathbf{q}}_{lg\_wlk} \right) \tag{4.52}$$

where ${}^{w}\mathbf{J}_{c\_lg}$ is the Jacobian mapping the support leg's joint velocities to the velocity of the camera, it is given by (4.44) where ${}^{b}\mathbf{t}_{re}$ is substituted by ${}^{b}\mathbf{t}_{c}$, such that ${}^{w}\mathbf{J}_{c\_b}\dot{\mathbf{q}}_{b\_wlk} = {}^{w}\mathbf{J}_{c\_lg} \cdot \dot{\mathbf{q}}_{lg\_wlk}$. From (4.52), and using the same exponential decrease of $\boldsymbol{e}(t)$, it can be shown that the compensated neck's joints velocities are

$$\dot{\mathbf{q}}_{nk\_wlk} = \widehat{\mathbf{J}}^{\dagger}_{e_{hd}} \left(-\Lambda_p \boldsymbol{e}(t) - \widehat{\mathbf{J}}_{e_{leg}} \cdot \dot{\mathbf{q}}_{lg\_wlk} \right) + \mathbf{P}_{e_{hd}} \cdot \mathbf{g}_{e_{hd}}(q) \tag{4.53}$$

where $\mathbf{J}_{e_{hd}} \triangleq \mathbf{L}_{\xi_{tsk}}\, {}^{c}\mathbf{W}_{w}\, {}^{w}\mathbf{J}_{c\_hd}$ and $\mathbf{J}_{e_{leg}} \triangleq \mathbf{L}_{\xi_{tsk}}\, {}^{c}\mathbf{W}_{w}\, {}^{w}\mathbf{J}_{c\_lg}$ are the sub-matrices of the visual task Jacobian, they are respectively associated with the neck and leg joints. $\widehat{\mathbf{J}}^{\dagger}_{e_{hd}} \in \mathbb{R}^{2\times k}$ is an estimate of the pseudo-inverse of $\mathbf{J}_{e_{hd}}$, $\mathbf{P}_{e_{hd}} \in \mathbb{R}^{2\times 2}$ is its associated projection operator, $\mathbf{g}_{e_{hd}}(q) \in \mathbb{R}^2$ is the joint velocities resulting from an auxiliary task. Thus, these joints velocities are compensated for all disturbances related to torso's motion along the z axis as well as its *roll* and *pitch* motions.

Remark. We have defined several variables with respect to an inertial frame denoted $\mathcal{F}_w$, when assuming a horizontal ground, this frame could be defined at the stance foot as in [161]. In a general case, the inertia measurement unit (IMU) can be used to determine the orientation but the translation needs a reference frame to be specified in the robot workspace.

4.4 Visual Servoing based Humanoid's Tracking

In a visual tracking task, a humanoid robot walks autonomously from an initial position towards a desired position relative to a moving target and, in contrast to positioning, the robot does not stop at that desired pose, but has to maintain it while walking.

This problem can be formulated similarly to that of positioning. However, an additional term accounting for the object's motion has to be included in the control design [15]. Thus, using the task function $\boldsymbol{e}(t)$ previously defined, its time derivative is

$$\dot{\boldsymbol{e}}(t) = \mathbf{J}_{e_{mb}} \dot{\mathbf{q}}_{mb} + \frac{\partial \boldsymbol{e}(t)}{\partial t} \tag{4.54}$$

where $\frac{\partial \boldsymbol{e}(t)}{\partial t} \neq 0$. Using control law (2.14) to generate the joints velocities, the close loop equation would be

$$\dot{\boldsymbol{e}}(t) = -\mathbf{J}_{e_{mb}} \widehat{\mathbf{J}}^{\dagger}_{e_{mb}} \Lambda_p \boldsymbol{e}(t) + \frac{\partial \boldsymbol{e}(t)}{\partial t} \tag{4.55}$$

Under the assumption that $\mathbf{J}_{e_{mb}} \widehat{\mathbf{J}}^{\dagger}_{e_{mb}} > 0$, the system (4.55) will converge, but with a steady state error function of $\frac{\partial \boldsymbol{e}(t)}{\partial t}$. If $\frac{\partial \boldsymbol{e}(t)}{\partial t}$ is constant, one may use an integral action [162]. In general a feed-forward action is often used to remove this steady state error [163, 26, 15]. Hence, the control law computing the joints motions while achieving an exponential decrease of the error can be written as follows

$$\begin{bmatrix} \dot{\mathbf{q}}_{nk} \\ \dot{\mathbf{q}}_{b} \end{bmatrix} = \dot{\mathbf{q}}_{mb} = \widehat{\mathbf{J}}^{\dagger}_{e_{mb}} \left(-\Lambda_p \boldsymbol{e}(t) - \widehat{\frac{\partial \boldsymbol{e}(t)}{\partial t}} \right) + \mathbf{P}_{e_{mb}} \cdot \mathbf{g}_{e_{mb}}(q) \tag{4.56}$$

where $\widehat{\frac{\partial \boldsymbol{e}(t)}{\partial t}}$ is an estimate of $\frac{\partial \boldsymbol{e}(t)}{\partial t}$.

Similarly to (4.53), the compensated neck's joint velocities will be given by

$$\dot{\mathbf{q}}_{nk\_wlk} = \widehat{\mathbf{J}}^{\dagger}_{e_{hd}} \left(-\Lambda_p \boldsymbol{e}(t) - \widehat{\frac{\partial \boldsymbol{e}(t)}{\partial t}} - \widehat{\mathbf{J}}_{e_{leg}} \cdot \dot{\mathbf{q}}_{lg\_wlk} \right) + \mathbf{P}_{e_{hd}} \cdot \mathbf{g}_{e_{hd}}(q) \tag{4.57}$$

Note that the method presented here is general and can be applied regardless of the chosen image features and their associated interaction matrix, one needs only to obtain an estimate of $\frac{\partial \boldsymbol{e}(t)}{\partial t}$. In what follows, we consider a particular case where an image point is used as a feature. This case is chosen in order to give a better insight in the tracking problem.

4.4.1 Image Feature Point based Tracking

Consider a moving target with a frame $\mathcal{F}_o$ attached to it in order to describe its motion. Let $P_i = (X_i, Y_i, Z_i)$ be a target point expressed in the camera frame $\mathcal{F}_c$ and $p_i = (x_i, y_i)$ its projection onto the image plane. Let the feature vector be defined as $\xi_i = [\ x_i \quad y_i\ ]^T$. Using (2.21), it can be shown that its time derivative could be written as

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_{\xi}{}^{c}\boldsymbol{V}_c + \underbrace{\begin{bmatrix} \frac{1}{Z_i} & 0 & -\frac{x_i}{Z_i} \\ 0 & \frac{1}{Z_i} & -\frac{y_i}{Z_i} \end{bmatrix}}_{-\mathbf{L}_{\xi_{(:,1:3)}}} \underbrace{{}^{c}\mathbf{R}_{w}\, {}^{w}\boldsymbol{v}_P}_{{}^{c}\boldsymbol{v}_P} \tag{4.58}$$

where $\mathbf{L}_\xi \in \mathbb{R}^{2\times 6}$ is the classical interaction matrix (2.22) associated with a feature point [30], ${}^c\boldsymbol{V}_c = \begin{bmatrix} {}^c\boldsymbol{\upsilon}_c^T & {}^c\boldsymbol{\omega}_c^T \end{bmatrix}^T$ is the camera's velocity screw expressed in the camera frame and ${}^c\boldsymbol{\upsilon}_P$ is the velocity of the target point expressed in $\mathcal{F}_c$. Given the fact that a point does not rotate, we can define ${}^c\boldsymbol{V}_o \triangleq \begin{bmatrix} {}^c\boldsymbol{\upsilon}_P^T & \mathbf{0}_{1\times 3} \end{bmatrix}^T$ as a pseudo screw velocity of the point in the camera frame. Using this definition, the time derivative of $\boldsymbol{e}(t)$ can be written as

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_\xi {}^c\boldsymbol{V}_c - \mathbf{L}_\xi {}^c\boldsymbol{V}_o \tag{4.59}$$

In Equation (4.59), $-\mathbf{L}_\xi {}^c\boldsymbol{V}_o = \frac{\partial \boldsymbol{e}(t)}{\partial t}$ and $\mathbf{L}_\xi {}^c\boldsymbol{V}_c = \mathbf{J}_{e_{mb}}\dot{\mathbf{q}}_{mb}$, with ${}^c\boldsymbol{V}_c$ being given by

$${}^c\boldsymbol{V}_c = {}^c\mathbf{W}_w {}^w\mathbf{J}_{mb}\dot{\mathbf{q}}_{mb} \tag{4.60}$$

From (4.59), assuming perfect knowledge of $\mathbf{L}_\xi$ and using control law (2.14), the camera velocity to yield an exponential decrease of the error and to track the target will be given by

$$\begin{aligned} {}^c\boldsymbol{V}_c &= \mathbf{L}_\xi^\dagger(-\Lambda_p \boldsymbol{e}(t) + \mathbf{L}_\xi {}^c\boldsymbol{V}_o) \\ &= {}^c\boldsymbol{V}_o - \mathbf{L}_\xi^\dagger \Lambda_p \boldsymbol{e}(t) \end{aligned} \tag{4.61}$$

where $\mathbf{L}_\xi^\dagger \mathbf{L}_\xi \approxeq \mathbf{I}$. If ${}^c\mathbf{J}_{mb} = {}^c\mathbf{W}_w {}^w\mathbf{J}_{mb}$ is defined as the robot Jacobian expressed in the camera frame. From (4.60) and (4.61), the joint velocities will be

$$\dot{\mathbf{q}}_{mb} = {}^c\mathbf{J}^\dagger{}_{mb} {}^c\boldsymbol{V}_o - {}^c\mathbf{J}^\dagger{}_{mb}\mathbf{L}_\xi^\dagger \Lambda_p \boldsymbol{e}(t) \tag{4.62}$$

after getting $\dot{\mathbf{q}}_{lg\_wlk}$ from the walking module, the compensated neck's joints velocities will be

$$\dot{\mathbf{q}}_{nk\_wlk} = {}^c\mathbf{J}^\dagger_{c\_hd} {}^c\boldsymbol{V}_o - {}^c\mathbf{J}^\dagger_{c\_hd}\left(\mathbf{L}_\xi^\dagger \Lambda_p \boldsymbol{e}(t) + {}^c\mathbf{J}_{c\_lg} \cdot \dot{\mathbf{q}}_{lg\_wlk}\right) \tag{4.63}$$

where the pseudo Jacobian ${}^c\mathbf{J}^\dagger_{c\_hd} = [{}^c\mathbf{W}_w {}^w\mathbf{J}_{c\_hd}]^\dagger$ and the Jacobian ${}^c\mathbf{J}_{c\_lg} = {}^c\mathbf{W}_w {}^w\mathbf{J}_{c\_lg}$.

Analyzing (4.63), gives important information on the tracking problem. We note that in the right hand side, the second term represents the velocity commands required to bring the camera from an initial pose to the desired relative pose, this is nothing but a positioning task, whose control action vanishes with the error $\boldsymbol{e}(t)$. On the other hand, the first term represents the control action necessary for the camera to follow the reference velocity imposed by the target. This velocity command is required to maintain the relative pose between the camera and the target. Note again that while ${}^c\boldsymbol{V}_o$ is the reference velocity in the camera space, ${}^c\mathbf{J}^\dagger{}_{mb} {}^c\boldsymbol{V}_o$ denotes its corresponding joints space reference velocity.

Given that in practice we only have approximate values of the Jacobian matrices and the target velocity, (4.62) and (4.63) will be rewritten as

$$\dot{\mathbf{q}}_{mb} = {}^{c}\widehat{\mathbf{J}}^{\dagger}{}_{mb}\left({}^{c}\widehat{\boldsymbol{V}}_{o} - \widehat{\mathbf{L}}_{\xi}^{\dagger}\Lambda_{p}\boldsymbol{e}(t)\right) \tag{4.64}$$

and

$$\dot{\mathbf{q}}_{nk\_wlk} = {}^{c}\widehat{\mathbf{J}}^{\dagger}_{c\_hd}\left[{}^{c}\widehat{\boldsymbol{V}}_{o} - \left(\widehat{\mathbf{L}}_{\xi}^{\dagger}\Lambda_{p}\boldsymbol{e}(t) + {}^{c}\widehat{\mathbf{J}}_{c\_lg}\cdot\dot{\mathbf{q}}_{lg\_wlk}\right)\right] \tag{4.65}$$

If good, the estimate of the target motion will keep the exponential decrease of the task function.

4.5 Visual Servoing based Humanoid's Grasping

We consider now grasping a target object as another robotic task to be performed while walking. The grasping task starts when the humanoid reaches the relative desired pose during either the positioning or the tracking task.

A visually guided grasping task while walking has already presented in [25], as compared to our application, the difference lies in the fact that here the walking is also visually servoed, while in the mentioned work the walking was predefined and the visually controlled grasping was implemented as secondary task in a task sequencing framework. The application presented here is a whole body task involving both legs joints for locomotion, the neck's joints for visual guidance and the arm joints for the grasping.

4.5.1 Modeling

It has been shown [164] that human uses 3D visual information to effectively solve the grasping problem. Similarly, we will use a PBVS scheme to model and control our task. For that purpose, let $\mathcal{F}_h$ denote a frame attached to the robot hand, $\mathcal{F}_o$ be the object's frame and $\mathcal{F}_c$ the camera frame (see Figure 4.6).

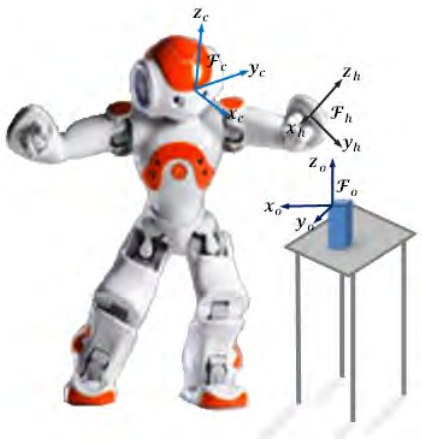


Figure 4.6: Coordinates frames definition for Grasping task

Using the visual measurements, the poses ${}^{c}\mathcal{X}_h$ and ${}^{c}\mathcal{X}_o \in \mathbb{R}^3 \times SO(3)$ respectively, of the hand and the object with respect to the camera can be estimated. For grasping, we can also define ${}^{o}\mathcal{X}_{h^d}$ as the desired pose of the hand with respect to the object. By defining a homogeneous transformation ${}^{j}H_i \in \mathbb{R}^{4\times4}$ corresponding to the pose ${}^{j}\mathcal{X}_i$, we can determine the following transformations

$$\begin{cases} {}^{h^d}H_h &= ({}^{c}H_o\,{}^{o}H_{h^d})^{-1}\,{}^{c}H_h \\ {}^{h}H_o &= ({}^{c}H_h)^{-1}\,{}^{c}H_o \\ {}^{h^d}H_o &= ({}^{o}H_{h^d})^{-1} \end{cases} \tag{4.66}$$

From transformations (4.66), we define the features vectors $\boldsymbol{\xi}_g(t) \triangleq ({}^{h}\mathbf{t}_o(t),\, \theta\mathbf{u}(t))$ and $\boldsymbol{\xi}_g^d \triangleq ({}^{h^d}\mathbf{t}_o,\, \mathbf{0})$, where ${}^{h}\mathbf{t}_o(t)$ and ${}^{h^d}\mathbf{t}_o \in \mathbb{R}^3$ denote the translation vectors from the object's frame to the current hand's frame and desired hand's frame, respectively. $\theta\mathbf{u}(t) \in \mathbb{R}^3$ is the angle/axis representation of the rotation ${}^{h^d}\mathbf{R}_h \in \mathbb{R}^{3\times3}$ between the current and desired hand's frame.

The task function being defined as $\boldsymbol{e}_g(t) \triangleq \boldsymbol{\xi}_g(t) - \boldsymbol{\xi}_g^d \in \mathbb{R}^6$, its time derivative is given by

$$\dot{\boldsymbol{e}}_g(t) = \begin{bmatrix} -{}^{h}\boldsymbol{\upsilon}_h - {}^{h}\boldsymbol{\omega}_h \times {}^{h}\mathbf{t}_o + {}^{h}\boldsymbol{\upsilon}_o \\ \mathbf{L}_{\theta\mathbf{u}}({}^{h}\boldsymbol{\omega}_h - {}^{h}\boldsymbol{\omega}_{h^d}) \end{bmatrix} \tag{4.67}$$

where ${}^{h}\boldsymbol{\upsilon}_h$ and ${}^{h}\boldsymbol{\omega}_h$ are respectively the linear and angular velocity vectors of the robot's hand expressed in the hand's frame $\mathcal{F}_h$, while ${}^{h}\boldsymbol{\upsilon}_o$ and ${}^{h}\boldsymbol{\omega}_{h^d}$ are respectively the linear velocity vector of the object and the angular velocity vector of the desired hand's frame with respect to the current hand's frame $\mathcal{F}_h$. $\mathbf{L}_{\theta\mathbf{u}}$ is given by [44]

$$\mathbf{L}_{\theta\mathbf{u}} = \mathbf{I}_3 - \frac{\theta}{2}[\mathbf{u}]_\times + \left(1 - \frac{sinc\,\theta}{sinc^2\frac{\theta}{2}}\right)[\mathbf{u}]_\times^2$$

Equation (4.67) can be rewritten as

$$\dot{\boldsymbol{e}}_g(t) = \underbrace{\begin{bmatrix} -\mathbf{I}_3 & [{}^{h}\mathbf{t}_o]_\times \\ \mathbf{0}_3 & \mathbf{L}_{\theta\mathbf{u}} \end{bmatrix}}_{\mathbf{L}_{\boldsymbol{e}_g}} \begin{bmatrix} {}^{h}\boldsymbol{\upsilon}_h \\ {}^{h}\boldsymbol{\omega}_h \end{bmatrix} + \underbrace{\begin{bmatrix} {}^{h}\mathbf{R}_w & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{L}_{\theta\mathbf{u}}\,{}^{h}\mathbf{R}_w \end{bmatrix} \begin{bmatrix} {}^{w}\boldsymbol{\upsilon}_o \\ {}^{w}\boldsymbol{\omega}_o \end{bmatrix}}_{\frac{\partial \boldsymbol{e}_g(t)}{\partial t}} \tag{4.68}$$

where $\mathbf{L}_{\boldsymbol{e}_g}$ is the interaction matrix associated with $\boldsymbol{e}_g(t)$, the products ${}^{h}\mathbf{R}_w\,{}^{w}\boldsymbol{\upsilon}_o \triangleq {}^{h}\boldsymbol{\upsilon}_o$ and ${}^{h}\mathbf{R}_w\,{}^{w}\boldsymbol{\omega}_o \triangleq {}^{h}\boldsymbol{\omega}_o = {}^{h}\boldsymbol{\omega}_{h^d}$ with $[\ {}^{w}\boldsymbol{\upsilon}_o^T \quad {}^{w}\boldsymbol{\omega}_o^T\ ]^T$ denoting the velocity screw of the object with respect to the inertial frame $\mathcal{F}_w$.

After expressing the hand's velocity screw $[\ {}^{h}\boldsymbol{\upsilon}_h^T \quad {}^{h}\boldsymbol{\omega}_h^T\ ]^T$ as function of the humanoid's joints velocities, (4.68) becomes

$$\dot{\boldsymbol{e}}_g(t) = \mathbf{L}_{\boldsymbol{e}_g}\,{}^{h}\mathbf{W}_w \underbrace{\begin{bmatrix} {}^{w}\mathbf{J}_{h\_arm} & {}^{w}\mathbf{J}_{h\_lg} \end{bmatrix}}_{{}^{w}\mathbf{J}_h} \begin{bmatrix} \dot{\mathbf{q}}_{arm\_wlk} \\ \dot{\mathbf{q}}_{lg\_wlk} \end{bmatrix} + \frac{\partial \boldsymbol{e}_g(t)}{\partial t} \tag{4.69}$$

where ${}^{w}\mathbf{J}_{h\_arm} \in \mathbb{R}^{6\times n_{arm}}$ and ${}^{w}\mathbf{J}_{h\_lg} \in \mathbb{R}^{6\times n_{leg}}$ are the Jacobians mapping, respectively, the arm's joints velocities ($\dot{\mathbf{q}}_{arm\_wlk} \in \mathbb{R}^{n_{arm}}$) and the support leg's joints velocities ($\dot{\mathbf{q}}_{lg\_wlk} \in \mathbb{R}^{n_{leg}}$) to the velocity screw of the hand's frame $\mathcal{F}_h$. ${}^{h}\mathbf{W}_{w}$ is the twist transformation matrix between $\mathcal{F}_w$ and $\mathcal{F}_h$. $\frac{\partial \boldsymbol{e}_g(t)}{\partial t}$, given in (4.68), denotes the variation of $\boldsymbol{e}_g(t)$ due to the object own motion.

4.5.2 Control

The control problem consists of computing the values of the joints velocities necessary to perform the task. Given that the leg's joints are already constrained by the locomotion module, the joints velocities to be computed are then those of the arm. Hence, If the object is motionless, the control law to compute the arm's joints motions and giving an exponential decrease of the error can be written as follows

$$\dot{\mathbf{q}}_{arm\_wlk} = \widehat{\mathbf{J}}^{\dagger}_{e_{g}\_arm}\left(-\Lambda_p \boldsymbol{e}(t) - \widehat{\mathbf{J}}_{e_{g}\_leg}\cdot\dot{\mathbf{q}}_{lg\_wlk}\right) + \mathbf{P}_{e_{g}\_arm}\cdot\mathbf{g}_{e_{arm}}(q) \tag{4.70}$$

where $\widehat{\mathbf{J}}^{\dagger}_{e_{g}\_arm}$ is the pseudo inverse of the estimate Jacobian of $\mathbf{J}_{e_{g}\_arm} \triangleq \mathbf{L}_{\boldsymbol{e}_g}\,{}^{h}\mathbf{W}_{w}\,{}^{w}\mathbf{J}_{h\_arm}$, $\mathbf{P}_{e_{g}\_arm}$ is its associated projection operator, $\mathbf{g}_{e_{arm}}(q)$ denotes the joints velocities resulting from a secondary task such as manipulability improvement or joint limits avoidance, and $\widehat{\mathbf{J}}_{e_{g}\_leg}$ is the estimate of Jacobian of $\mathbf{J}_{e_{g}\_leg} \triangleq \mathbf{L}_{\boldsymbol{e}_g}\,{}^{h}\mathbf{W}_{w}\,{}^{w}\mathbf{J}_{h\_lg}$.

In general, when the target is moving, $\frac{\partial \boldsymbol{e}(t)}{\partial t} \neq 0$, the control law (4.70) can be rewritten as

$$\dot{\mathbf{q}}_{arm\_wlk} = \widehat{\mathbf{J}}^{\dagger}_{e_{g}\_arm}\left(-\Lambda_p \boldsymbol{e}_g(t) - \widehat{\frac{\partial \boldsymbol{e}_g(t)}{\partial t}} - \widehat{\mathbf{J}}_{e_{g}\_leg}\cdot\dot{\mathbf{q}}_{lg\_wlk}\right) + \mathbf{P}_{e_{arm}}\cdot\mathbf{g}_{e_{arm}}(q) \tag{4.71}$$

where $\widehat{\frac{\partial \boldsymbol{e}(t)}{\partial t}}$ is an estimate of $\frac{\partial \boldsymbol{e}(t)}{\partial t}$, which could be obtained, for example, using (4.68) if the object velocity is known. A disturbance observer can also be used. Indeed, the grasping task starts only when the robot reaches the desired pose with respect to the target; since this relative pose is supposed to be fixed, any object's own motion, from the grasping point of view, is regarded as a disturbance.

The manipulability improvement, as additional task, can concern either the arm alone or both the arm and the "moving base" similarly to the case of wheeled mobile manipulator [127]. In the first case, the Jacobian in the manipulability measure is limited to that of the arm, while in the second case, the latter Jacobian is extended with that of the base.

4.6 Conclusion

In this chapter, we have formulated a theoretical framework allowing visual servoing of a humanoid robot. We have seen that finding an appropriate task Jacobian for a biped robot is not as straightforward as for manipulators or to some extent wheeled robots. Indeed, computed from the instantaneous base (the stance foot), the Jacobian will yield joints velocities incompatible with locomotion constraints imposed by the structure and bipedal locomotion of the robot. We have then reformulated the kinematics of the active chain, using the floating base concept. In this way, we managed to separate continuous from discrete dynamics of this hybrid system, allowing thus to define easily continuous task Jacobian and to integrate walking with all its related constraints. It is worth noting that this general formulation is not limited to bipedal walking; it can also be extended to crawling for example. One would simply need to substitute for the appropriate locomotion module.

In dealing with the redundancy problem due to the robot's high number of DoFs and extended workspace, we have illustrated how auxiliary tasks such as aligning the humanoid's sight and walking directions, avoiding joint limits can be performed and also how the manipulability index of the robot can be optimized. Finally, we have formulated visual control schemes for applications such as positioning, tracking and grasping, which will be experimentally validated in the following chapter.

Chapter 5

Experimental Validation of Visual Servoing on Humanoid NAO

This chapter applies the visual servoing framework developed in the previous chapter to the humanoid robot NAO V4.0 [144] used as experimental platform in this study. First, the robot is presented and then its kinematics model derived (forward, inverse, and velocity kinematics). Once the model is obtained, the simulations are carried out and then follows the experiments on the actual robot. Finally, the chapter ends with a conclusion.

5.1 Presentation of Humanoid NAO

We consider the humanoid robot NAO, which is a small size, full actuated biped robot manufactured by Aldebaran Robotics [144]. The version under study - V4.0 - has a total of 25 degrees of freedom (DoFs), is 58 cm tall, and about 5 kg of mass. The structure of NAO consists of a head with two joints, two symmetrical upper and lower limbs with six and five joints respectively, and a trunk with one joint in its pelvis. In each of NAO's arms, 2 DoFs are located at the shoulder, 2 DoFs at the elbow, and 1 DoF at the wrist, while an additional DoF is assigned to the hand for grasping. Each leg has 2 DoFs located at the hip, 1 DoF at the knee and 2 DoFs at the ankle. Unlike other humanoid robots, the degree of freedom of the pelvis is shared between the two legs via a special mechanism [165, 166]. Figure 5.1 shows all the 25 DoFs as mounted and labeled on NAO Robot.

NAO's hardware platform is equipped with different sensors: two cameras, four directional microphones, sonar rangefinder (from 0.01 m to 3 m), two infra red (IR) emitters and receivers, one inertial measurement unit (comprising a two axis gyroscope and a three axis accelerometer), nine tactile sensors and eight pressure sensors [144]. The cameras are identical and embedded in its head. The "top camera", located in the forehead, provides a view of the horizon and the "bottom camera" located in the mouth and directed downward, senses the immediate surrounding of the

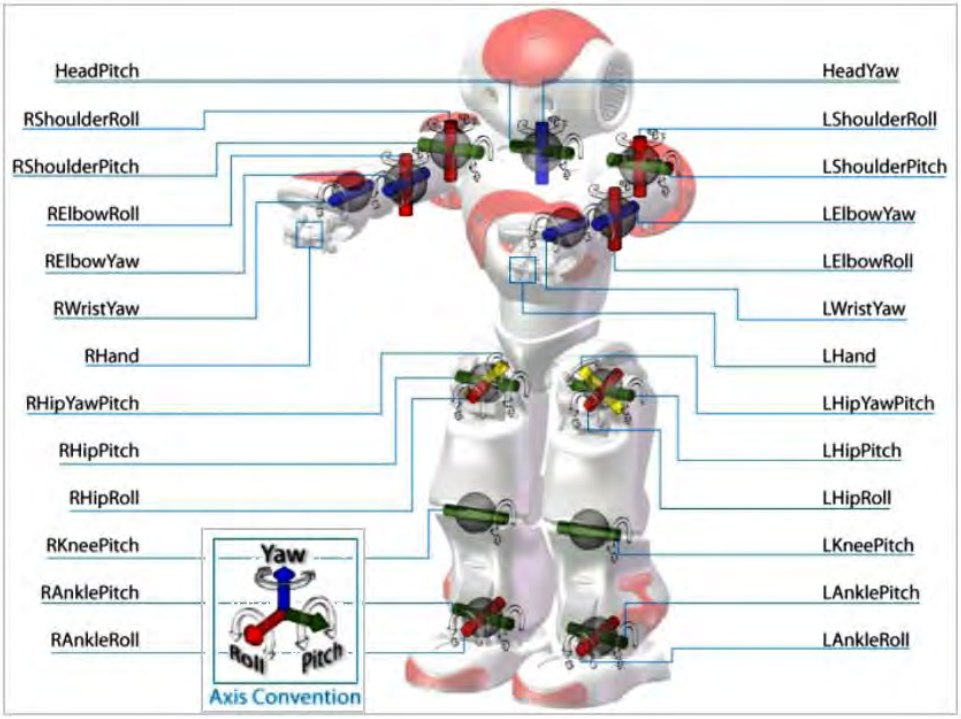


Figure 5.1: Humanoid robot NAO V 4.0 with all its degrees of freedom labeled [1]

robot. Both cameras can take up to 30 frames per second. They have fixed focus ranging from 30 cm to infinite and their vertical and horizontal fields of view are respectively of 47.6° and 60.9°. The pressure sensors, located at the sole of each foot (four per foot), can be used to compute the position of the robot's center of pressure (CoP or ZMP), which an important indicator of balance and stability.

NAO robot has an omnidirectional walking ability exploiting a simplified dynamic model with an LMPC based pattern generator [167]. The joints sensors information and the CoP position are used in feedback to stabilize the robot about the generated walking trajectories. This improves the robustness of the walking to small disturbances and allows the robot to walk on different surfaces (tiles, carpet, etc.).

From a software point of view, NAO is a fully programmable robot, its programming framework, NAOqi, is cross-platform and cross-language. NAOqi can be used on Windows, Linux or Mac platforms and applications can be developed in Python or in C++. Moreover, the programming framework offers a graphical user interface, Choregraphe, which eases the creation of applications and various forms of behaviors with different level of complexity thanks to a set of pre-programmed functions (ref. [168] for more details).

5.2 Kinematic Modeling of Humanoid Robot NAO

In reference to the definitions given in Section 3.1.1, the humanoid NAO can be modeled as a kinematic tree with mobile base [169] located in the torso and five serial chains: the head, the two arms and the two legs [170].

However, the kinematics analysis can be very complex because of the high number of degrees of freedom. in order to systematically assign frames and carry out this analysis, the Denavit-Hartenberg (D-H) convention [126, p. 64] is generally used (see Appendix B.1).

5.2.1 D-H Convention based Frames Assignment on NAO

Assigning frames to NAO robot is then equivalent to assigning frames to its five kinematic chains sharing the torso as common base. Following D-H convention, after defining first a fixed inertial frame $\mathcal{F}_W = (O_w, \overrightarrow{x}_w, \overrightarrow{y}_w, \overrightarrow{z}_w)$, and the moving frames $\mathcal{F}_B = (O_B, \overrightarrow{x}_B, \overrightarrow{y}_B, \overrightarrow{z}_B)$ and $\mathcal{F}_{iE} = (O_{iE}, \overrightarrow{x}_{iE}, \overrightarrow{y}_{iE}, \overrightarrow{z}_{iE})$ $(i = 1 \dots 5)$ respectively attached to the base and to each chain's end-effector, frames are assigned to every link as depicted in Figure 5.2.

Using this frames assignment, the kinematic modeling can be carried out. We only focus on internal configuration, leaving out the transformation from $\mathcal{F}_B$ to $\mathcal{F}_w$ which depends on coordinates external to the robot. Also, the results in the text are limited to those of the head, the left arm and the left leg chains; the kinematics of the right arm and the right leg are given in appendix B. All modeling results have been obtained using symbolic mathematics toolbox of ***Matlab***.

5.2.2 Forward kinematics of NAO

In deriving the forward kinematics of NAO, we seek to express all its end-effectors positions and orientations as function of the joints variables. Using the D-H parametrization, this is achieved by concatenating homogeneous transformations between frames assigned to adjacent links of each chain. The resulting transformation can be written as

$$\mathbf{T}_{iE}^{w} \triangleq \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

where the vector $\boldsymbol{p} = [\ p_x \quad p_y \quad p_z\ ]^T$ represents the position of EE_i with respect to $\mathcal{F}_w$ and the vectors $\boldsymbol{n} = [\ n_x \quad n_y \quad n_z\ ]^T$, $\boldsymbol{o} = [\ o_x \quad o_y \quad o_z\ ]^T$, and $\boldsymbol{a} = [\ a_x \quad a_y \quad a_z\ ]^T$ are respectively called the *normal*, *orientation* or *sliding*, and *approach* vectors of EE_i [126, p. 69]. The vectors $\boldsymbol{n}$, $\boldsymbol{o}$ and $\boldsymbol{a}$ form the rotation matrix ${}^{w}\mathbf{R}_{iE} \in \mathbb{R}^{3\times3}$ of EE_i with respect to $\mathcal{F}_w$.

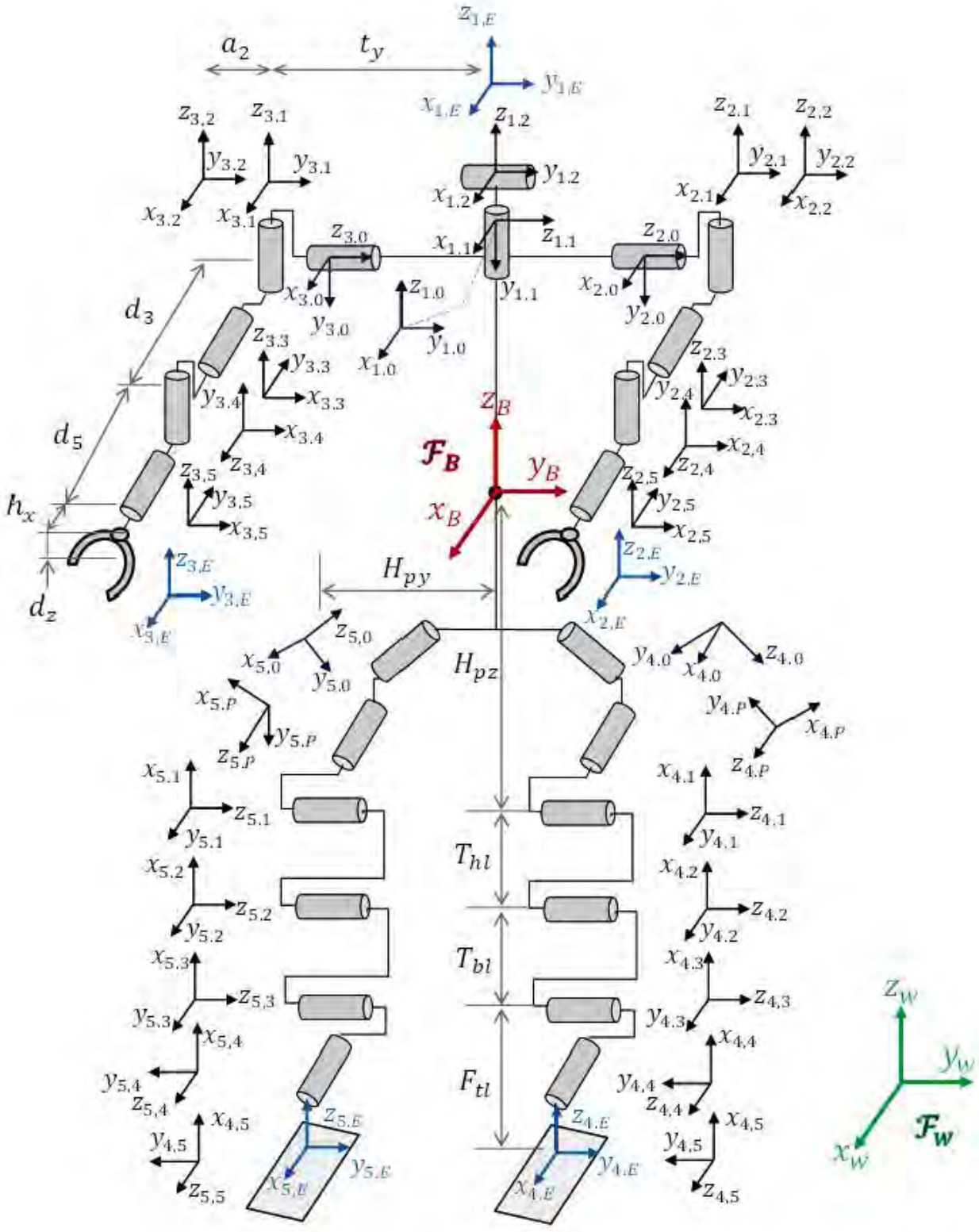


Figure 5.2: Frames assignment used for kinematic analysis of humanoid NAO

5.2.2.1 Forward kinematics of NAO's Head

According to frames assignment in Figure 5.2, the D-H parameters of NAO robot's head, with one of the camera taken as end-effector, are summarized in table (5.1).

Table 5.1: DH. parameters of NAO's Head

Link (joint)	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$	***Range (degrees)***
Base	T(0,0,NeckOffsetZ)				
HeadYaw	0	-90°	0	θ_1	-119.5-119.5
HeadPitch	0	90°	0	θ_2	-38.5-29.5
End Effector	R(90).R(90).T(0,he,we)				

where *he* and *we* are the camera (end-effector) offsets with respect to the neck (see Figure 5.2). Substituting these parameters in the D-H matrix (B.1) gives the following matrices

$$A^0_{11} = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A^1_{12} = \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

where c_i and s_i refer to $cos\theta_i$ and $sin\theta_i$, respectively. The transformation matrix from the frame $O_0x_0y_0z_0$ to the base frame $O_Bx_By_Bz_B$ and the transformation matrix from the end-effector's frame $O_Ex_Ey_Ez_E$ to the frame $O_2x_2y_2z_2$ are respectively given by

$$A^B_{10} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A^2_{1E} = \begin{bmatrix} 0 & 0 & 1 & we \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & he \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

where d_1 represents the distance between the frames $O_0x_0y_0z_0$ and $O_Bx_By_Bz_B$ along z_B-axis.

The final orientation and position of the head's end-effector frame with respect to the base frame are obtained from the transformation $\mathbf{T}^B_{1E}$, which is computed by multiplying the homogenous transformations (5.2) and (5.3) as follows

$$T^B_{1E} = A^B_{10}A^0_{11}A^1_{12}A^2_{1E} \tag{5.4}$$

which, finally, results in

$$\mathbf{T}^E_{1E} = \begin{bmatrix} -s_1 & c_1s_2 & c_1s_2 & c_1c_2w_e + c_1s_2h_e \\ c_1 & s_1s_2 & s_1c_2 & s_1c_2w_e + s_1s_2h_e \\ 0 & c_2 & -s_2 & -s_2w_e + c_2h_e + d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.5}$$

5.2.2.2 Forward kinematics of NAO's Left Arm

Similarly to the head, we derive the D-H parameters of NAO's left hand from the frame assignment in Figure 5.2 and summarized them in table (5.2).

Table 5.2: DH. parameters of NAO's Left Arm

Link (joint)	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$	***Range (degrees)***
Base	T_{ZB}(ShouldderOffsetZ).T_{YB}(ShoulderOffsetY).R_{XB}(-90°)				
LshoulderPitch	0	90°	0	θ_1	-119.5-119.5
LshoulderRoll	a_2	90°	0	$\theta_2 + 90$	-18 - 76
LElbowYaw	0	-90°	d_3	θ_3	-119.5-119.5
LElbowRoll	0	90°	0	θ_4	-88.5 - 2
LWristYaw	0	-90°	d_5	θ_5	-104 - 104.2
End Effector	R_{Z2}(-90).T_{Z5}(-HandOffsetZ).T_{Y5}(-HandOffsetY)				

where a_2, d_3, and d_5 are respectively the elbow lateral offset, the upper arm length, and the hand horizontal offset (see Figure B.1). Substituting table (5.2)'s parameters in the D-H matrix (B.1) gives the following matrices

$$A^0_{21} = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ A^1_{22} = \begin{bmatrix} -s_2 & 0 & c_2 & a_2 s_2 \\ c_2 & 0 & s_2 & a_2 c_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ A^2_{23} = \begin{bmatrix} c_3 & 0 & -s_3 & 0 \\ s_3 & 0 & c_3 & 0 \\ 0 & -1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A^3_{24} = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ A^4_{25} = \begin{bmatrix} c_5 & 0 & -s_5 & 0 \\ s_5 & 0 & c_5 & 0 \\ 0 & -1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.6}$$

The transformation matrix from the frame $O_0x_0y_0z_0$ to the base frame $O_Bx_By_Bz_B$ and the transformation matrix from the end-effector's frame $O_Ex_Ey_Ez_E$ to the frame $O_5x_5y_5z_5$ to are respectively given by

$$A^B_{20} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & t_y \\ 0 & -1 & 0 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix},\ A^5_{2E} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -h_x \\ 0 & 0 & 1 & -d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.7}$$

where t_y and t_z denote the shoulder offsets along the y_B-axis and along the z_B-axis, respectively; while h_x and d_z denote the hand frame's offsets along the x_5-axis and along the z_5-axis, respectively. Finally, the homogeneous transformation $\mathbf{T}^B_{2E}$ giving the orientation and position of the left hand's end-effector frame with respect to the base frame is obtained from the product of homogenous transformations (5.6) and (5.7) as follows

$$\mathbf{T}^B_{2E} = A^B_{20} A^0_{21} A^1_{22} A^2_{23} A^3_{24} A^4_{25} A^5_{2E} \tag{5.8}$$

Using the notation (5.1), we obtain

$$n_2 = \begin{bmatrix} (-c_1 s_2 c_3 + s_1 s_3) s_4 + c_1 c_2 c_4 \\ c_2 c_3 s_4 + s_2 c_4 \\ (s_1 s_2 c_3 + c_1 s_3) s_4 - c_2 c_4 \end{bmatrix} \tag{5.9}$$

$$o_2 = \begin{bmatrix} ((-c_1 s_2 c_3 + s_1 s_3) c_4 - c_1 c_2 s_4) c_5 + (c_1 s_2 s_3 + s_1 c_3) s_5 \\ (c_2 c_3 c_4 - s_2 s_4) c_5 - c_2 s_3 s_5 \\ ((s_1 s_2 c_3 + c_1 s_3) c_4 + s_1 c_2 s_4) c_5 + (-s_1 s_2 s_3 + c_1 c_3) s_5 \end{bmatrix} \tag{5.10}$$

$$a_2 = \begin{bmatrix} ((c_1 s_2 c_3 - s_1 s_3) c_4 + c_1 c_2 s_4) s_5 + (c_1 s_2 s_3 + s_1 c_3) c_5 \\ (-c_2 c_3 c_4 + s_2 s_4) s_5 - c_2 s_3 c_5 \\ -((s_1 s_2 c_3 + c_1 s_3) c_4 + s_1 c_2 s_4) s_5 + (-s_1 s_2 s_3 + c_1 c_3) c_5 \end{bmatrix} \tag{5.11}$$

$$p_2 = \begin{bmatrix} \{((-c_1 s_2 c_3 + s_1 s_3) s_4 + c_1 c_2 c_4)(h_x + d_5) + (((-c_1 s_2 c_3 + s_1 s_3) c_4 - c_1 c_2 s_4) s_5 \\ \quad -(c_1 s_2 s_3 + s_1 c_3) c_5) d_z + c_1 (c_2 d_3 - s_2 a_2)\} \\ \{(c_2 c_3 s_4 + s_2 c_4)(h_x + d_5) + ((c_2 c_3 c_4 - s_2 s_4) s_5 + c_2 s_3 c_5) d_z + s_2 d_3 + c_2 a_2 + t_y\} \\ \{((s_1 s_2 c_3 + c_1 s_3) s_4 - s_1 c_2 c_4)(h_x + d_5) + (((s_1 s_2 c_3 + c_1 s_3) c_4 + s_1 c_2 s_4) s_5 \\ \quad -(-s_1 s_2 s_3 + c_1 c_3) c_5) d_z - s_1 (c_2 d_3 + s_2 a_2) + t_z\} \end{bmatrix} \tag{5.12}$$

5.2.2.3 Forward kinematics of NAO's Left Leg

NAO's legs have five joints each and are connected to the base (torso) frame via the pelvis joint, which makes chains of six joints linking each foot to the torso frame as depicted in Figure 5.2. Hence, from the frames assigned to the left leg's chain, the D-H parameters of NAO's left leg are derived and summarized in table (5.3).

Table 5.3: DH. parameters of NAO's Left Leg

Link (joint)	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$	***Range (degrees)***
Base	T_{ZB}(HipOffsetZ).T_{YB}(HipOffsetY).R_{XB}(-135°)				
LHipYawPitch	0	-90°	0	$\theta_p - 90^o$	-65.62 - 42.44
LHipRoll	0	90°	0	$\theta_1 + 45^o$	-21.74 - 45.29
LHipPitch	$-T_{hl}$	0	0	θ_2	-101.63 - 27.73
LKneePitch	$-T_{bl}$	0	0	θ_3	-5.29 - 121.04
LAnklePitch	0	-90°	0	θ_4	-68.5 - 52.86
LAnkleRoll	$-F_{tl}$	0	0	θ_5	-22.79 - 44.06
End Effector	$R_{X5}(90^o).R_{Y5}(90^o)$				

In this table T_{hl}, T_{bl}, and F_{tl} are respectively the thigh length, the tibia length, and the foot height. Substituting table (5.3)'s parameters in the D-H matrix (B.1) gives the following matrices

$$A_{41}^{P} = \begin{bmatrix} c_{1+45^o} & 0 & s_{1+45^o} & 0 \\ s_{1+45^o} & 0 & -c_{1+45^o} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{42}^{1} = \begin{bmatrix} c_2 & -s_2 & 0 & -T_{hl}c_2 \\ s_2 & c_2 & 0 & -T_{hl}s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{43}^{2} = \begin{bmatrix} c_3 & -s_3 & 0 & -T_{bl}c_3 \\ s_3 & c_3 & 0 & -T_{bl}s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$A_{44}^{3} = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{45}^{4} = \begin{bmatrix} c_5 & -s_5 & 0 & -T_{bl}c_5 \\ s_5 & c_5 & 0 & -T_{bl}s_5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.13}$$

where c_{i+45^o}, and s_{i+45^o} refer to $cos(\theta_i + 45^o)$ and $sin(\theta_i + 45^o)$ respectively, while the superscript $_P$ stands for pelvis. The transformation matrix from the frame $O_0x_0y_0z_0$ to the base frame $O_Bx_By_Bz_B$, the transformation matrix from the pelvis frame $O_Px_Py_Pz_P$ to the frame $O_0x_0y_0z_0$, and the transformation matrix from the end-effector's frame $O_Ex_Ey_Ez_E$ to the frame $O_5x_5y_5z_5$ are respectively given by

$$A_{40}^{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{-135^o} & -s_{-135^o} & H_{py} \\ 0 & s_{-135^o} & c_{-135^o} & -H_{pz} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{4P}^{0} = \begin{bmatrix} s_p & 0 & c_p & 0 \\ -c_p & 0 & s_p & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{4E}^{5} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.14}$$

where H_{py} and H_{pz} denote the hip offsets along the y_B-axis and along the z_B-axis, respectively. Finally, the homogeneous transformation $\mathbf{T}_{4E}^{B}$ giving the orientation and position of the left leg's end-effector frame with respect to the base coordinate frame is obtained from the product of homogenous transformations (5.13) and (5.14) as follows

$$\mathbf{T}_{4E}^{B} = A_{40}^{B} A_{4P}^{0} A_{41}^{P} A_{42}^{1} A_{43}^{2} A_{44}^{3} A_{45}^{4} A_{4E}^{5} \tag{5.15}$$

Using the definition (5.1) and the notation s_{ijk} , respectively c_{ijk} for $sin(\theta_i + \theta_j + \theta_k)$, respectively $cos(\theta_i + \theta_j + \theta_k)$, we obtain

$$n_4 = \begin{bmatrix} -s_p c_{1+45^o} s_{234} + c_p c_{234} \\ -\frac{\sqrt{2}}{2}(c_p c_{1+45^o} - s_{1+45^o}) s_{234} - \frac{\sqrt{2}}{2} s_p c_{234} \\ -\frac{\sqrt{2}}{2}(c_p c_{1+45^o} + s_{1+45^o}) s_{234} - \frac{\sqrt{2}}{2} s_p c_{234} \end{bmatrix} \tag{5.16}$$

$$o_4 = \begin{bmatrix} (c_p c_{1+45^o} c_{234} + c_p s_{234}) s_5 + s_p s_{1+45^o} c_5 \\ -\frac{\sqrt{2}}{2}((c_p c_{1+45^o} - s_{1+45^o}) c_{234} - \frac{\sqrt{2}}{2} s_p s_{234} s_5) + \frac{\sqrt{2}}{2}(s_p c_{1+45^o} + c_{1+45^o}) c_5 \\ -\frac{\sqrt{2}}{2}((c_p c_{1+45^o} + s_{1+45^o}) c_{234} - \frac{\sqrt{2}}{2} s_p s_{234} s_5) + \frac{\sqrt{2}}{2}(s_p c_{1+45^o} - c_{1+45^o}) c_5 \end{bmatrix} \tag{5.17}$$

$$a_4 = \begin{bmatrix} (s_p c_{1+45^o} c_{234} + c_p s_{234}) c_5 - s_p s_{1+45^o} s_5 \\ (\frac{\sqrt{2}}{2}((c_p c_{1+45^o} - s_{1+45^o}) c_{234} - \frac{\sqrt{2}}{2} s_p s_{234} c_5) - \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) s_5 \\ (\frac{\sqrt{2}}{2}((c_p c_{1+45^o} + s_{1+45^o}) c_{234} - \frac{\sqrt{2}}{2} s_p s_{234} c_5) - \frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) s_5 \end{bmatrix} \tag{5.18}$$

$$p_4 = \begin{bmatrix} \{-s_p[c_{1+45^o}(T_{bl} c_{23} + T_{hl} c_2 + F_{tl} c_{234} c_5) - s_{1+45^o} F_{tl} s_5] \\ \quad -c_p(T_{bl} s_{23} + T_{hl} s_2 + F_{tl} s_{234} c_5)\} \\ \{-\frac{\sqrt{2}}{2}[(c_p c_{1+45^o} - s_{1+45^o})(T_{bl} c_{23} + T_{hl} c_2 + F_{tl} c_{234} c_5) \\ \quad -s_p(T_{bl} s_{23} + T_{hl} s_2 + F_{tl} s_{234} c_5) - (c_p s_{1+45^o} + c_{1+45^o}) F_{tl} s_5] + H_{py}\} \\ \{-\frac{\sqrt{2}}{2}[(c_p c_{1+45^o} + s_{1+45^o})(T_{bl} c_{23} + T_{hl} c_2 + F_{tl} c_{234} c_5) \\ \quad -s_p(T_{bl} s_{23} + T_{hl} s_2 + F_{tl} s_{234} c_5 - (c_p s_{1+45^o} - c_{1+45^o}) F_{tl} s_5] - H_{pz}\} \end{bmatrix} \tag{5.19}$$

5.2.3 Inverse Kinematics of NAO

The inverse kinematics consists of finding the joints values θ_j that will yield the given end-effector's pose. In our case, for a given position $\boldsymbol{p}_i = [\ p_{x_i} \quad p_{y_i} \quad p_{z_i}\ ]^T$ and orientation $\boldsymbol{\phi}_i = [\ \alpha_i \quad \beta_i \quad \gamma_i\ ]^T$ (parametrized by Euler angles, for instance) of the i^{th} end-effector with respect to the robot's base $\mathcal{F}_b$ frame located in the torso, we will seek close form solution(s) to the inverse kinematics. We start by computing first the associated homogeneous transformation matrix $\mathbf{T}_{iE}^b \in \mathbb{R}^{4\times4}$ as

$$T_{iE}^b \triangleq \begin{bmatrix} {}^b\mathbf{R}_{iE} & \boldsymbol{p}_i \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \tag{5.20}$$

where ${}^b\mathbf{R}_{iE} \in \mathbb{R}^{3\times3}$ denotes the rotation matrix associated with $\boldsymbol{\phi}_i$ and defined as

$${}^b\mathbf{R}_{iE} \triangleq [\ \boldsymbol{n}_i \quad \boldsymbol{o}_i \quad \boldsymbol{a}_i\ ] = \begin{bmatrix} n_{x_i} & o_{x_i} & a_{x_i} \\ n_{y_i} & o_{y_i} & a_{y_i} \\ n_{z_i} & o_{z_i} & a_{z_i} \end{bmatrix}, \tag{5.21}$$

with $\boldsymbol{n}_i$, $\boldsymbol{o}_i$, and $\boldsymbol{a}_i$ representing respectively the normal, the orientation and the approach vectors. Once obtained, this transformation is used, in conjunction with the forward kinematics, to solve the problem.

5.2.3.1 Inverse Kinematics of NAO's Head

Let us assume that the position $\boldsymbol{p}_1 = [\ p_{1x} \quad p_{1y} \quad p_{1z}\ ]^T$ of the head's end-effector (camera) is given. By identification with (5.5), we can write

$$\begin{cases} p_{1x} & = cos\theta_1(w_e cos\theta_2 + h_e sin\theta_2) \\ p_{1y} & = sin\theta_1(w_e cos\theta_2 + h_e sin\theta_2)\ . \\ p_{1z} & = -w_e sin\theta_2 + h_e cos\theta_2 + d_1 \end{cases} \tag{5.22}$$

The joint angle θ_1 can be deduce from the first two row of (5.22) as

$$\theta_1 = atan\left(\frac{p_{1y}}{p_{1x}}\right). \tag{5.23}$$

To compute θ_2, the third row of (5.22) is rewritten as

$$p_{1z} - d_1 = -w_e sin\theta_2 + h_e cos\theta_2. \tag{5.24}$$

By defining $tan\varphi \triangleq \frac{h_e}{-w_e}$, it can be shown that (5.24) can be written as

$$sin(\theta_2 + \varphi) = \frac{-p_z + d_1}{\sqrt{w_e^2 + h_e^2}}. \tag{5.25}$$

Hence, we obtain θ_2 as follows

$$\theta_2 = asin\left(\frac{-p_z + d_1}{\sqrt{w_e^2 + h_e^2}}\right) - arctan\left(\frac{h_e}{-w_e}\right). \tag{5.26}$$

If instead, the orientation ($\boldsymbol{n}_1$ $\boldsymbol{o}_1$ $\boldsymbol{a}_1$) is provided, it can be shown that the neck's joints angles would be

$$\theta_1 = atan\left(-\frac{n_x}{n_y}\right) \tag{5.27}$$

$$\theta_2 = atan\left(-\frac{a_z}{o_z}\right) \tag{5.28}$$

5.2.3.2 Inverse Kinematics of NAO's Left Arm

Consider a given pose of the left hand's end-effector, written in form of homogenous transformation $\mathbf{T}_{2E}^{B}$, its expression as function of the arm joints is given by Equation (5.8), which is recalled here

$$\mathbf{T}_{2E}^{B} = A_{20}^{B} A_{21}^{0} A_{22}^{1} A_{23}^{2} A_{24}^{3} A_{25}^{4} A_{2E}^{5}. \tag{5.29}$$

Computation of θ_1 and θ_2. To find θ_1 and θ_2, let us first compute $\mathbf{T}_{24}^{2}$ by post-multiplying $\mathbf{T}_{2E}^{B}$ by $(A_{2E}^{5})^{-1}(A_{25}^{4})^{-1}$ and pre-multiplying by $(A_{20}^{B})^{-1}$, we obtain

$$\mathbf{T}_{24}^{0} = (A_{20}^{B})^{-1}\mathbf{T}_{2E}^{B}(A_{2E}^{5})^{-1}(A_{25}^{4})^{-1} = A_{21}^{0} A_{22}^{1} A_{23}^{2} A_{24}^{3}. \tag{5.30}$$

The last columns of both sides of (5.30) give the following equality

$$\begin{cases} -n_x(d_5 + h_x) + a_x d_z + p_x & = cos\theta_1(d_3 cos\theta_2 - a_2 sin\theta_2) \\ n_z(d_5 + h_x) - a_z d_z - p_z + t_z & = sin\theta_1(d_3 cos\theta_2 - a_2 sin\theta_2) \\ -n_y(d_5 + h_x) + a_y d_z + p_y - t_y & = d_3 sin\theta_2 + a_2 cos\theta_2 \end{cases} \tag{5.31}$$

Hence, the first two row of (5.31) give θ_1 as

$$\theta_1 = atan\left(\frac{n_z(d_5+h_x) - a_z d_z - p_z + t_z}{-n_x(d_5+h_x) + a_x d_z + p_x}\right) \pm 180^o \tag{5.32}$$

By considering now the third row of (5.31) and defining $tan\varphi \triangleq \frac{a_2}{d_3}$, it can be shown that

$$sin(\theta_2+\varphi) = \frac{-n_y(d_5+h_x) + a_y d_z + p_y - t_y}{\sqrt{a_2^2+d_3^2}}, \tag{5.33}$$

which gives θ_2 as follows

$$\theta_2 = asin\left(\frac{-n_y(d_5+h_x) + a_y d_z + p_y - t_y}{\sqrt{a_2^2+d_3^2}}\right) - atan\left(\frac{a_2}{d_3}\right). \tag{5.34}$$

Computation of θ_3 and θ_4. To find θ_3 and θ_4, let us now compute $\mathbf{T}_{25}^2$ by post-multiplying $\mathbf{T}_{2E}^B$ by $(A_{2E}^5)^{-1}$ and pre-multiplying by $(A_{21}^0)^{-1}(A_{20}^B)^{-1}\mathbf{T}_{2E}^B(A_{2E}^5)^{-1}$, we obtain

$$\mathbf{T}_{25}^2 = (A_{22}^1)^{-1}(A_{21}^0)^{-1}(A_{20}^B)^{-1}\mathbf{T}_{2E}^B(A_{2E}^5)^{-1} = A_{23}^2 A_{24}^3 A_{25}^4 \tag{5.35}$$

From the second column of both sides of (5.35), we can write

$$\begin{cases} sin\theta_2(n_x cos\theta_1 - n_z sin\theta_1) - n_y cos\theta_2 & = -cos\theta_3 sin\theta_4 \\ -n_x sin\theta_1 - n_z cos\theta_1 & = -sin\theta_3 sin\theta_4 \\ -cos\theta_2(n_x cos\theta_1 - n_z sin\theta_1) - n_y sin\theta_2 & = -cos\theta_4 \end{cases} \tag{5.36}$$

NAOWith θ_1 and θ_2 already known, θ_3 and θ_4 can be computed as follows

$$\theta_3 = atan\left(\frac{-n_x sin\theta_1 + n_z cos\theta_1}{sin\theta_2(n_x cos\theta_1 + n_z sin\theta_1) - n_y cos\theta_2}\right), \tag{5.37}$$

$$\theta_4 = -acos\left(cos\theta_2(n_x cos\theta_1 - n_z sin\theta_1) + n_y sin\theta_2\right). \tag{5.38}$$

Computation of θ_5. To find θ_5, we equate the third rows elements of both sides of (5.35) and consider only the resulting first and third expression, we have

$$\begin{cases} cos\theta_2(o_x cos\theta_1 - o_z sin\theta_1) + o_y sin\theta_2 & = -sin\theta_4 cos\theta_5 \\ cos\theta_2(a_x cos\theta_1 - a_z sin\theta_1) + a_y sin\theta_2 & = sin\theta_4 sin\theta_5 \end{cases} \tag{5.39}$$

Hence, θ_5 is obtained as follows

$$\theta_5 = atan\left(-\frac{cos\theta_2(a_x cos\theta_1 + a_z sin\theta_1) + a_y sin\theta_2}{cos\theta_2(o_x cos\theta_1 + o_z sin\theta_1) + o_y sin\theta_2}\right) \tag{5.40}$$

5.2.3.3 Inverse Kinematics of NAO's Left Leg

Solving the inverse kinematics of NAO's legs is not straightforward as for the head or the arms. Indeed, the hip-yaw joints of the two legs are mechanically connected and driven by a single motor [166]. Because of these constraints, the inverse kinematics of six dimensional poses specified for both legs cannot be realized simultaneously (we would have 12 constraints for 11 DoFs).

Thus, the solution presented here, assumes a single support phase and gives priority to the stance foot by relaxing the pose constraints on the swing leg. In this way, the stance leg's inverse kinematics can be solved as that of a six DoFs serial chain with spherical joints [124], [126].

Consider the left leg as stance leg, with a given pose of its end-effector written in form of homogenous transformation $\mathbf{T}_{4E}^{B}$. For the reader convenience, let us rewrite the expression of $\mathbf{T}_{4E}^{B}$ as function of the leg joints given by Equation (5.8),

$$\mathbf{T}_{4E}^{B} = A_{40}^{B} A_{41}^{0} A_{41}^{P} A_{42}^{1} A_{43}^{2} A_{44}^{3} A_{45}^{4} A_{4E}^{5} \tag{5.41}$$

Computation of θ_3. Let us start by θ_3, which can be computed by applying the law of cosines on the triangle formed by the thigh, the tibia, and the line drawn between the hip joints and the ankle joints (see Figure 5.3).

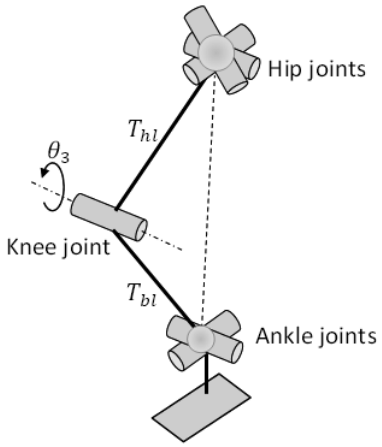


Figure 5.3: Leg serial chain of humanoid NAO

Let us compute $\mathbf{T}_{43}^{1}$ by post-multiplying $\mathbf{T}_{4E}^{B}$ by $(A_{4E}^{5})^{-1}(A_{45}^{4})^{-1}(A_{44}^{3})^{-1}$ and pre-multiplying the result by $(A_{41}^{P})^{-1}(A_{4P}^{0})^{-1}(A_{40}^{B})^{-1}$, we obtain

$$\mathbf{T}_{43}^{1} = (A_{41}^{P})^{-1}(A_{4P}^{0})^{-1}(A_{40}^{B})^{-1}\mathbf{T}_{4E}^{B}(A_{4E}^{5})^{-1}(A_{45}^{4})^{-1}(A_{44}^{3})^{-1} = A_{42}^{1}A_{43}^{2} \tag{5.42}$$

Squaring the moduli of position vectors of both sides of (5.42) gives

$$(p_x + a_x F_{tl})^2 + (p_x + a_y F_{tl} - H_{py})^2 + (p_z + a_z F_{tl} + H_{pz})^2 = T_{bl}^2 + 2T_{bl}T_{hl}cos\theta_3 + T_{hl}^2, \quad (5.43)$$

from where we compute θ_3 as follows

$$\theta_3 = acos\left(\frac{(p_x + a_x F_{tl})^2 + (p_x + a_y F_{tl} - H_{py})^2 + (p_z + a_z F_{tl} + H_{pz})^2}{2T_{bl}T_{hl}}\right) \quad (5.44)$$

Computation of θ_4 and θ_5. In order to find θ_4 and θ_5, consider the inverse serial chain (i.e. from the foot up to the torso frame), within which the position is determined by the joints θ_3, θ_4 and θ_5, while the orientation by the joints θ_P, θ_1 and θ_2. With this consideration, we can use geometrical decoupling [126] between the position and the orientation and solve them independently.

Let us start by computing $\mathbf{T}_{41}^5$, which is the inverse of $\mathbf{T}_{45}^1$ obtained by post-multiplying $\mathbf{T}_{4E}^B$ by $(A_{4E}^5)^{-1}$ and pre-multiplying the result by $(A_{41}^P)^{-1}(A_{4P}^0)^{-1}(A_{40}^B)^{-1}$, hence we have

$$\begin{aligned}\mathbf{T}_{41}^5 &= \left((A_{41}^P)^{-1}(A_{4P}^0)^{-1}(A_{40}^B)^{-1}\mathbf{T}_{4E}^B(A_{4E}^5)^{-1}\right)^{-1} = \left(A_{42}^1 A_{43}^2 A_{44}^3 A_{45}^4\right)^{-1} \\ &= (A_{45}^4)^{-1}(A_{44}^3)^{-1}(A_{43}^2)^{-1}(A_{42}^1)^{-1}\end{aligned} \quad (5.45)$$

From the equality of the last columns of both sides of (5.45), we can write

$$\begin{cases} {}_1^5P_x &= \frac{p_x(n_z o_y - n_y o_z) + (p_y - H_{py})(n_x o_z - n_z o_x) + (p_z + H_{pz})(n_y o_x - n_x o_y)}{n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y} \\ {}_1^5P_y &= \frac{p_x(n_z a_y - n_y a_z) + (p_y - H_{py})(n_x a_z - n_z a_x) + (p_z + H_{pz})(n_y a_x - n_x a_y)}{n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y} \\ {}_1^5P_z &= \frac{p_x(o_z a_y - o_y a_z) + (p_y - H_{py})(o_x a_z - o_z a_x) + (p_z + H_{pz})(o_y a_x - o_x a_y)}{n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y} \end{cases} \quad (5.46)$$

and

$$\begin{cases} {}_1^5P_x &= cos\theta_5[T_{hl}cos(\theta_3 + \theta_4) + T_{bl}cos\theta_4] + F_{tl} \\ {}_1^5P_y &= -sin\theta_5[T_{hl}cos(\theta_3 + \theta_4) + T_{bl}cos\theta_4] \\ {}_1^5P_z &= -[T_{hl}sin(\theta_3 + \theta_4) + T_{bl}sin\theta_4] \end{cases} \quad (5.47)$$

Hence, as long as $T_{hl}cos(\theta_3 + \theta_4) + T_{bl}cos\theta_4 \neq 0$, from (5.46) and (5.47), θ_5 will be given by

$$\theta_5 = atan\left(\frac{-[p_x(n_z a_y - n_y a_z) + (p_y - H_{py})(n_x a_z - n_z a_x) + (p_z + H_{pz})(n_y a_x - n_x a_y)]}{p_x(n_z o_y - n_y o_z) + (p_y - H_{py})(n_x o_z - n_z o_x) + (p_z + H_{pz})(n_y o_x - n_x o_y) - F_{tl}D_r}\right) \quad (5.48)$$

where D_r is the determinant of the rotation matrix ${}^B\mathbf{R}_{4E}$, extracted from $\mathbf{T}_{4E}^B$, and given by

$$D_r \triangleq n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y \quad (5.49)$$

Given that θ_3 is already known, θ_4 can be computed from the third equation of (5.47), which is rewritten as

$$ {}^{5}_{1}P_z = -sin\theta_4(T_{hl}cos\theta_3 + T_{bl}) - cos\theta_4(T_{hl}sin\theta_3) \tag{5.50} $$

After using the trigonometric properties $sin\theta_4 = \frac{tan\theta_4}{\sqrt{1+tan^2\theta_4}}$ and $cos\theta_4 = \frac{1}{\sqrt{1+tan^2\theta_4}}$, it can be shown that

$$ \theta_4 = atan\left(\frac{-(T_{hl}cos\theta_3 + T_{bl})T_{hl}sin\theta_3 \pm {}^{5}_{1}P_z\sqrt{T_{hl}^2 + T_{bl}^2 + 2cos\theta_3 T_{hl}T_{bl} - ({}^{5}_{1}P_z)^2}}{(T_{hl}cos\theta_3 + T_{bl})^2 - ({}^{5}_{1}P_z)^2}\right) \tag{5.51} $$

Computation θ_P, θ_1 and θ_2. To find θ_P, θ_1 and θ_2, let us start by finding T^0_{42}, obtained by by post-multiplying $\mathbf{T}^B_{4E}$ by $(A^5_{4E})^{-1}(A^4_{45})^{-1}(A^3_{44})^{-1}(A^2_{43})^{-1}$ and pre-multiplying the result by $(A^B_{40})^{-1}$. Hence,

$$ \mathbf{T}^0_{42} = (A^B_{40})^{-1}\mathbf{T}^B_{4E}(A^5_{4E})^{-1}(A^4_{45})^{-1}(A^3_{44})^{-1}(A^2_{43})^{-1} \tag{5.52} $$

From the third columns of the both sides of (5.52), we obtain the following set of equations

$$ \begin{cases} -a_x sin\theta_5 + o_x cos\theta_5 & = sin\theta_p sin(\theta_1 + 45^o) \\ \frac{\sqrt{2}}{2}[sin\theta_5(a_y + a_z) - cos\theta_5(o_y + o_z)] & = -cos\theta_p sin(\theta_1 + 45^o) \\ -\frac{\sqrt{2}}{2}[sin\theta_5(a_y - a_z) - cos\theta_5(o_y - o_z)] & = cos(\theta_1 + 45^o) \end{cases} \tag{5.53} $$

Given that θ_5 is already known, considering the first two rows of (5.53), it can be shown that θ_P will be given by

$$ \theta_P = atan\left(\frac{a_x tan\theta_5 - o_x}{\frac{\sqrt{2}}{2}[tan\theta_5(a_y + a_z) - (o_y + o_z)]}\right) \tag{5.54} $$

Similarly, from the third row of (5.53), we obtain θ_1 as follows

$$ \theta_1 = acos\left(\frac{\sqrt{2}}{2}[cos\theta_5(o_y - o_z) - sin\theta_5(a_y - a_z)]\right) - \frac{\pi}{4} \tag{5.55} $$

Finally, θ_2 can be found by equating the third rows of both sides of Equation (5.52), and taking the first two equalities. That is

$$ \begin{cases} \frac{\sqrt{2}}{2}\,(cos(\theta_3 + \theta_4)[cos\theta_5(a_z - a_y) + sin\theta_5(o_z - o_y)] + sin(\theta_3 + \theta_4)(n_y - n_z)) & = -sin(\theta_1 + \frac{\pi}{4})cos\theta_2 \\ \frac{\sqrt{2}}{2}\,(sin(\theta_3 + \theta_4)[cos\theta_5(a_y - a_z) + sin\theta_5(o_y - o_z)] + cos(\theta_3 + \theta_4)(n_y - n_z)) & = sin(\theta_1 + \frac{\pi}{4})sin\theta_2 \end{cases}. \tag{5.56} $$

Thus, from (5.56) it can be shown that

$$ \theta_2 = atan\left(\frac{tan(\theta_3 + \theta_4)[cos\theta_5(a_y - a_z) + sin\theta_5(o_y - o_z)] + (n_y - n_z)}{[cos\theta_5(a_z - a_y) + sin\theta_5(o_z - o_y)] + tan(\theta_3 + \theta_4)(n_y - n_z)}\right) \tag{5.57} $$

$$ \theta_4 = atan\left(\frac{-(T_{hl}cos\theta_3 + T_{bl})T_{hl}sin\theta_3 \pm {}^{5}_{1}P_z\sqrt{T_{hl}^2 + T_{bl}^2 + 2cos\theta_3 T_{hl}T_{bl} - ({}^{5}_{1}P_z)^2}}{(T_{hl}cos\theta_3 + T_{bl})^2 - ({}^{5}_{1}P_z)^2}\right) \tag{5.58} $$

5.2.4 Velocity Kinematics

Thus, using definitions and notations given in Section 3.1.1.3, in order to determine velocity kinematics of NAO, we will derive for each chain, relatively to the base, the Jacobian Matrix mapping joints velocities to end-effectors velocities.

Considering only the internal joint $\mathbf{q}_h$, once the forward kinematics has been determined, it can be shown [126] that $\mathbf{J}_i(\mathbf{q}_h)$ can be written as $\mathbf{J}_i(\mathbf{q}_h) = \begin{bmatrix} J_{i_1} & J_{i_2} & \dots & J_{i_{N_j}} \end{bmatrix}$, with J_{i_j} obtained as follows

$$J_{i_j} = \begin{bmatrix} z_{j-1} \times (o_{iE} - o_{j-1}) \\ z_{j-1} \end{bmatrix} \tag{5.59}$$

where z_{j-1} and o_{j-1} can be derived from the transformation $\mathbf{T}^B_{j-1}$, respectively as the first three elements of the third and fourth column. z_{j-1} can also be obtained as

$$z_{j-1} = {}^B\mathbf{R}_{j-1}\boldsymbol{k} \tag{5.60}$$

where ${}^B\mathbf{R}_{j-1}$ is the rotation matrix in $\mathbf{T}^B_{j-1}$ and $\boldsymbol{k}$ is the unit vector associated with $\overrightarrow{z}_B$. The resulting J_{i_j} is also expressed with respect to $\mathcal{F}_B$.

5.2.4.1 NAO Head's Jacobian

Consider NAO's head kinematic chain, which has two intersecting joints ($o_0 = o_1$). Let us denote the common frames origin by the vector $\boldsymbol{o}$ expressed in the base frame, given that all joints are revolute, it can be shown [126] that the head's Jacobian has the form

$$J_{head} = \begin{bmatrix} z_0 \times (o_E - \boldsymbol{o}) & z_1 \times (o_E - \boldsymbol{o}) \\ z_0 & z_1 \end{bmatrix} \tag{5.61}$$

Using the forward kinematics derived in section 5.2.2, from the transformation $\mathbf{T}^B_{1E}$, $\mathbf{T}^B_{10}$, and $\mathbf{T}^B_{11}$ obtained by multiplying the A-matrices (5.2)-(5.3), it can be shown that the position vectors of the end-effector $\boldsymbol{o}_E$ and the origin $\boldsymbol{o}$ are given by

$$o_E = \begin{bmatrix} c_1c_2w_e + c_1s_2h_e \\ s_1c_2w_e + s_1s_2h_e \\ -s_2w_e + c_2h_e + d_1 \end{bmatrix}, \qquad o = \begin{bmatrix} 0 \\ 0 \\ d_1 \end{bmatrix}. \tag{5.62}$$

Using the rotation matrices $\mathbf{R}^B_{10}$, and $\mathbf{R}^B_{11}$ extracted from the same homogenous transformations, the joints axes of rotation are computed according to (5.60) and are given by

$$z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad z_1 = \begin{bmatrix} -s_1 \\ c_1 \\ 0 \end{bmatrix} \tag{5.63}$$

Substituting (5.62) and (5.63) in (5.61) yields

$$J_{head} = \begin{bmatrix} -s_1(c_2 w_e + s_2 h_e) & -c_1(s_2 w_e - c_2 h_e) \\ c_1(c_2 w_e + s_2 h_e) & -s_1(s_2 w_e - c_2 h_e) \\ 0 & -c_2 w_e - s_2 h_e \\ 0 & -s_1 \\ 0 & c_1 \\ 1 & 0 \end{bmatrix} \tag{5.64}$$

5.2.4.2 NAO Left Arm's Jacobian

Consider NAO left arm kinematic chain, with its five joints whose two intersect at the shoulder and two at the elbow. All joints being revolute, the Jacobian will be obtained as follows

$$\mathbf{J}_{L\_Arm} = \begin{bmatrix} z_0 \times (o_E - o_0) & z_1 \times (o_E - o_1) & z_2 \times (o_E - o_2) & z_3 \times (o_E - o_3) & z_4 \times (o_E - o_4) \\ z_0 & z_1 & z_2 & z_3 & z_4 \end{bmatrix}, \tag{5.65}$$

where the o_i and z_i are all expressed with respect to the base frame (torso) and computed using the forward kinematics. Hence, from $\mathbf{T}_{2E}^{B}$ the position vector of the left arm's end-effector is given by

$$o_E = \begin{bmatrix} [((-c_1 s_2 c_3 + s_1 s_3)s_4 + c_1 c_2 c_4)(h_x + d_5) + (((-c_1 s_2 c_3 + s_1 s_3)c_4 - c_1 c_2 s_4)s_5 \\ -(c_1 s_2 s_3 + s_1 c_3)c_5)d_z + c_1(c_2 d_3 - s_2 a_2)] \\ [(c_2 c_3 s_4 + s_2 c_4)(h_x + d_5) + ((c_2 c_3 c_4 - s_2 s_4)s_5 + c_2 s_3 c_5)d_z + s_2 d_3 + c_2 a_2 + t_y] \\ -[((s_1 s_2 c_3 + c_1 s_3)s_4 - s_1 c_2 c_4)(h_x + d_5) - \{((s_1 s_2 c_3 + c_1 s_3)c_4 + s_1 c_2 s_4)s_5 \\ +(s_1 s_2 s_3 - c_1 c_3)c_5\}d_z + s_1(c_2 d_3 - s_2 a_2) + t_z] \end{bmatrix}, \tag{5.66}$$

and from homogenous transformations $\mathbf{T}_{20}^{B}$, $\mathbf{T}_{21}^{B}$, $\mathbf{T}_{22}^{B}$, $\mathbf{T}_{23}^{B}$ and $\mathbf{T}_{24}^{B}$, derived from the product of A-matrices in (5.6)-(5.7), the positions of the frames' origins are obtained as

$$o_0 = o_1 = \begin{bmatrix} 0 \\ t_y \\ t_z \end{bmatrix},\ o_2 = \begin{bmatrix} -a_2 c_1 s_2 \\ a_2 c_2 + t_y \\ -a_2 s_1 s_2 + t_z \end{bmatrix},\ \text{and } o_3 = o_4 = \begin{bmatrix} d_3 c_1 c_2 - a_2 c_1 s_2 \\ d_3 s_2 + a_2 c_2 + t_y \\ d_3 s_1 c_2 - a_2 s_1 s_2 + t_z \end{bmatrix}, \tag{5.67}$$

while using the extracted rotation matrices $\mathbf{R}_{20}^{B}$, $\mathbf{R}_{21}^{B}$, $\mathbf{R}_{22}^{B}$, $\mathbf{R}_{23}^{B}$ and $\mathbf{R}_{24}^{B}$ and Equation (5.60), the joints axes of rotation are obtained as

$$z_0 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},\ z_1 = \begin{bmatrix} s_1 \\ 0 \\ -c_1 \end{bmatrix},\ z_2 = \begin{bmatrix} c_1 c_2 \\ s_2 \\ s_1 c_2 \end{bmatrix},\ z_3 = \begin{bmatrix} c_1 s_2 s_3 + s_1 c_3 \\ -c_2 s_3 \\ s_1 s_2 s_3 - c_1 c_3 \end{bmatrix}, \tag{5.68}$$

$$z_4 = \begin{bmatrix} (-c_1 s_2 s_3 + s_1 c_3)s_4 + c_1 c_2 c_4 \\ c_2 c_3 c_4 + s_2 c_4 \\ (-s_1 s_2 s_3 - c_1 c_3)s_4 + s_1 c_2 c_4 \end{bmatrix} \tag{5.69}$$

Substituting (5.66)-(5.69) in (5.65) gives the column vectors of $\mathbf{J}_{L\_Arm}$ as follows

$$J_{LA11} = \begin{bmatrix} \{-(s_1s_2s_3 + c_1c_3)[(h_x + d_5)s_4 + d_z s_5 c_4] \\ +s_1[d_3c_2 - a_2s_2 - d_z(s_5c_2s_4 + c_5s_2s_3) + (h_x + d_5)c_2c_4] + d_z c_5 c_1 c_3]\} \\ 0 \\ \{(c_1s_2c_3 - s_1s_3)[(h_x + d_5)s_4 + d_z s_5 c_4] \\ +c_1[-d_3c_2 + a_2s_2 + d_z(s_5c_2s_4 + c_5s_2s_3) - (h_x + d_5)c_2c_4] + d_z c_5 c_1 c_3\} \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$z_4 = \begin{bmatrix} (-c_1s_2s_3 + s_1c_3)s_4 + c_1c_2c_4 \\ c_2c_3c_4 + s_2c_4 \\ (-s_1s_2s_3 - c_1c_3)s_4 + s_1c_2c_4 \end{bmatrix} \tag{5.70}$$

$$J_{LA12} = \begin{bmatrix} c_1\{(h_x + d_5)(c_2c_3s_4 + s_2c_4) + d_z[s_5(c_2c_3c_4 - s_2s_4) + c_2s_3c_5] + d_3c_2 + a_2s_2\} \\ (h_x + d_5)(s_2c_3s_4 - c_2c_4) + d_z[s_5(s_2c_3c_4 + c_2s_4) + s_2s_3c_5] - d_3c_2 + a_2s_2 \\ s_1\{(h_x + d_5)(c_2c_3s_4 + s_2c_4) + d_z[s_5(c_2c_3c_4 - s_2s_4) + c_2s_3c_5] + d_3c_2 + a_2s_2\} \\ s_1 \\ 0 \\ -c_1 \end{bmatrix}$$

$$J_{LA13} = \begin{bmatrix} -(c_1s_2s_3 + s_1c_3)[(h_x + d_5)s_4 + d_z c_5 c_4] + d_z c_5(c_1s_2c_3 - s_1s_3)] \\ c_2[(h_x + d_5)s_3s_4 + d_z(s_5c_4s_3 - c_5c_3)] \\ -(s_1s_2s_3 - s_1c_3)[(h_x + d_5)s_4 + d_z s_5 c_4] + d_z c_5(s_1s_2c_3 + c_1s_3)] \\ c_1c_2 \\ s_2 \\ s_1c_2 \end{bmatrix}$$

$$J_{LA14} = \begin{bmatrix} (c_1s_2s_3 - s_1c_3)[(h_x + d_5)c_4 - d_z s_5 s_4] + c_1c_2[(h_x + d_5)s_4 + d_z s_5 c_4] \\ c_2c_3[-(h_x + d_5)c_4 + d_z s_5 s_4] + s_2[(h_x + d_5)s_4 + d_z s_5 c_4] \\ (s_1s_2c_3 + c_1s_3)[(h_x + d_5)c_4 - d_z s_5 s_4] + s_1c_2[(h_x + d_5)s_4 + d_z s_5 c_4] \\ c_1s_2s_3 + s_1c_3 \\ -c_2s_3 \\ s_1s_2s_3 - c_1c_3 \end{bmatrix}$$

$$J_{LA14} = \begin{bmatrix} -d_z[c_5c_4(s_1s_3 - c_1s_2c_3) + s_5(s_1c_3 + c_1s_2s_3) + c_5s_4c_1c_2] \\ d_z[c_5(s_2s_4 - c_2c_3c_4) + s_5c_2s_3] \\ d_z[c_5c_4(c_1s_3 + s_1s_2c_3) + s_5(c_1c_3 - s_1s_2s_3) + c_5s_4s_1c_2] \\ (-c_1s_2s_3 + s_1c_3)s_4 + c_1c_2c_4 \\ c_2c_3c_4 + s_2c_4 \\ (-s_1s_2s_3 - c_1c_3)s_4 + s_1c_2c_4 \end{bmatrix}$$

Finally, $\mathbf{J}_{LArm}$ can be written as

$$\mathbf{J}_{LArm} = \begin{bmatrix} J_{LA11} & J_{LA12} & J_{LA13} & J_{LA14} & J_{LA15} \end{bmatrix} \tag{5.71}$$

5.2.4.3 NAO Left Leg's Jacobian

Consider the kinematic chain of NAO left leg, from the foot to the torso frame via the pelvis. This chain has six joints with three of them intersecting at the hip and two at the ankle. All joints being revolute, the Jacobian can be computed as follows

$$\mathbf{J}_{LLeg} = \begin{bmatrix} z_0 \times (o_E - o_0) & z_P \times (o_E - o_P) & z_1 \times (o_E - o_1) & z_2 \times (o_E - o_2) & z_3 \times (o_E - o_3) & z_4 \times (o_E - o_4) \\ z_0 & z_p & z_1 & z_2 & z_3 & z_4 \end{bmatrix} \tag{5.72}$$

Using the forward kinematics, from $\mathbf{T}_{4E}^{B}$, the position of the left leg's end-effector is

$$o_E = \begin{bmatrix} \{-s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_{1+45^o}F_{tl}s_5] - c_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5)\} \\ \{-\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} - s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ -(c_ps_{1+45^o} + c_{1+45^o})F_{tl}s_5] + H_{py}\} \\ \{-\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} + s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ -(c_ps_{1+45^o} - c_{1+45^o})F_{tl}s_5] - H_{pz}\} \end{bmatrix} \tag{5.73}$$

The position vectors of the frames origins are derived from the homogenous transformations $\mathbf{T}_{40}^{B}$, $\mathbf{T}_{4P}^{B}$, $\mathbf{T}_{41}^{B}$, $\mathbf{T}_{42}^{B}$, $\mathbf{T}_{43}^{B}$ and $\mathbf{T}_{44}^{B}$ (derived from (5.13) - (5.14)) as follows

$$o_0 = o_P = o_1 = \begin{bmatrix} 0 \\ H_{py} \\ -H_{pz} \end{bmatrix},\ o_2 = \begin{bmatrix} -(s_pc_{1+45^o}c_2 + c_ps_2)T_{hl} \\ -\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} - s_{1+45^o})c_2 - s_ps_2]T_{hl} + H_{py} \\ -\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} + s_{1+45^o})c_2 - s_ps_2]T_{hl} - H_{pz} \end{bmatrix}, \text{ and}$$

$$o_3 = o_4 = \begin{bmatrix} -s_pc_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2) - c_p(T_{bl}s_{23} + T_{hl}s_2) \\ -\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} - s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2) - s_p(T_{bl}s_{23} + T_{hl}s_2)] + H_{py} \\ -\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} + s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2) - s_p(T_{bl}s_{23} + T_{hl}s_2)] - H_{pz} \end{bmatrix}. \tag{5.74}$$

Using Equation (5.60) and the rotation matrices $\mathbf{R}_{40}^{B}$, $\mathbf{R}_{4P}^{B}$, $\mathbf{R}_{41}^{B}$, $\mathbf{R}_{42}^{B}$, $\mathbf{R}_{43}^{B}$ and $\mathbf{R}_{44}^{B}$, the joints' axes of rotation are obtained as

$$z_0 = \begin{bmatrix} 0 \\ \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix},\ z_1 = \begin{bmatrix} c_p \\ -\frac{\sqrt{2}}{2}s_p \\ -\frac{\sqrt{2}}{2}s_p \end{bmatrix},\ z_1 = z_2 = z_3 = \begin{bmatrix} s_ps_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_ps_{1+45^o} + c_{1+45^o}) \\ \frac{\sqrt{2}}{2}(c_ps_{1+45^o} - c_{1+45^o}) \end{bmatrix},$$

$$z_4 = \begin{bmatrix} -s_ps_{1+45^o}s_{234} + c_pc_{234} \\ -\frac{\sqrt{2}}{2}[s_{234}(c_pc_{1+45^o} - s_{1+45^o}) + s_pc_{234}] \\ -\frac{\sqrt{2}}{2}[s_{234}(c_pc_{1+45^o} + s_{1+45^o}) + s_pc_{234}] \end{bmatrix} \tag{5.75}$$

Substituting (5.73)-(5.75) in (5.72) gives the column vectors of $\mathbf{J}_{LLeg}$ as follows

$$J_{LL11} = \begin{bmatrix} -c_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_{1+45^o}F_{tl}s_5] + s_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ \frac{\sqrt{2}}{2}\{s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_{1+45^o}F_{tl}s_5] + c_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5)\} \\ \frac{\sqrt{2}}{2}\{s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_{1+45^o}F_{tl}s_5] + c_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5)\} \\ 0 \\ \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix} \tag{5.76}$$

$$J_{LL12} = \begin{bmatrix} s_p[s_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + c_{1+45^o}F_{tl}s_5] \\ \frac{\sqrt{2}}{2}[(c_ps_{1+45^o} + c_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + (c_pc_{1+45^o} - s_{1+45^o})s_5F_{tl}] \\ \frac{\sqrt{2}}{2}[(c_ps_{1+45^o} - c_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + (c_pc_{1+45^o} + s_{1+45^o})s_5F_{tl}] \\ c_p \\ -\frac{\sqrt{2}}{2}s_p \\ -\frac{\sqrt{2}}{2}s_p \end{bmatrix} \tag{5.77}$$

$$J_{LL13} = \begin{bmatrix} s_p s_{1+45^o}(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) - c_p(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} - c_{1+45^o})(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5)] \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5)] \\ s_p s_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) \end{bmatrix} \tag{5.78}$$

$$J_{LL14} = \begin{bmatrix} s_p s_{1+45^o}(T_{bl}s_{23} + F_{tl}s_{234}c_5) - c_p(T_{bl}c_{23} + F_{tl}c_{234}c_5) \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} - c_{1+45^o})(T_{bl}s_{23} + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + F_{tl}c_{234}c_5)] \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})(T_{bl}s_{23} + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + F_{tl}c_{234}c_5)] \\ s_p s_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) \end{bmatrix} \tag{5.79}$$

$$J_{LL15} = \begin{bmatrix} (s_p c_{1+45^o} s_{234} - c_p c_{234})F_{tl}c_5 \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} - c_{1+45^o})s_{234} + s_p c_{234}]F_{tl}c_5 \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})s_{234} + s_p c_{234}]F_{tl}c_5 \\ s_p s_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) \end{bmatrix} \tag{5.80}$$

$$J_{LL16} = \begin{bmatrix} [(s_p c_{1+45^o} c_{234} + c_p s_{234})s_5 + s_p s_{1+45^o} c_5]F_{tl} \\ \frac{\sqrt{2}}{2}\{[(c_p s_{1+45^o} - c_{1+45^o})c_{234} - s_p s_{234}]s_5 + (c_p s_{1+45^o} + c_{1+45^o})c_5]F_{tl} \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})c_{234} - s_p s_{234}]s_5 + (c_p s_{1+45^o} - c_{1+45^o})c_5]F_{tl} \\ -s_p s_{1+45^o} s_{234} + c_p c_{234} \\ -\frac{\sqrt{2}}{2}[s_{234}(c_p c_{1+45^o} - s_{1+45^o}) + s_p c_{234}] \\ -\frac{\sqrt{2}}{2}[s_{234}(c_p c_{1+45^o} + s_{1+45^o}) + s_p c_{234}] \end{bmatrix} \tag{5.81}$$

Finally, $\mathbf{J}_{LLeg}$ can be written as

$$\mathbf{J}_{LLeg} = \begin{bmatrix} J_{LL11} & J_{LL12} & J_{LL13} & J_{LL14} & J_{LL15} & J_{LL16} \end{bmatrix} \tag{5.82}$$

The Jacobian matrices which have just been determined are key elements in deriving the robot dynamics and also in visual servoing formulation as seen the previous chapters.

5.3 Simulation Results on Visual Servoing of Humanoid NAO

This section validates through simulations the visual servoing framework developed for humanoid robots. Applying the proposed framework on humanoid NAO, three kind of autonomous tasks will be simulated: positioning, tracking and grasping.

5.3.1 Simulation Setup

The overall kinematic model, which has just been determined has been validated and used to build a 3D wire-frame model of the humanoid able to simulate the robot's behavior as the latter walks or executes different tasks. The lower part of this model featured already in Section 3.4. In this section, the full model will be used to simulate visual servoing on humanoid NAO. The Matlab simulator we have written is based on the reactive omnidirectional walking pattern generator developed in Chapter 3, the visual servoing framework proposed in Chapter 4 and the humanoid NAO model derived in this Chapter. All these codes and their dependencies are given in the attached CD. The camera featuring on the model is NAO's top camera. IBVS is employed as the main technique to drive the humanoid robot. The target is a square with four coplanar points (side $L = 0.20\,m$). The interaction matrix is a weighted sum of the current and the desired one $(L_{s_{tsk}} = (1-\rho).L_s + \rho.L_{s^*})$, where ρ was set to 0.2. It is chosen to prevent ill conditioning or singularity during execution of the task [34]. The depths of the feature points are assumed to be roughly known. The gain Λ was set to 0.3.

In all subsequent simulations, the robot starts at the origin of axes with its top camera located at $0.465\,m$ from the ground and with an orientation of $(\frac{\pi}{2},\, 0.0,\, \frac{\pi}{2})\,rad$, such that its z_c axis is normal to the YZ plane and its y_c axis pointing in the Z direction.

In order to evaluate the execution of a given task, different variables are plotted and analyzed:

- the features error, which has to converge towards zero for the task to be completed.
- the trajectories of the feature points. These are the state variables in the image space; they must remain defined in the image, otherwise the servoing will stop.
- the 3D trajectory of the camera. Resulting from IBVS, it is required to be acceptable.
- the velocity commands generated by the visual servoing algorithm. They should stay within the range of the robot's joints and should decrease towards zero as the task converges.
- the global displacement of the robot, this shows the humanoid's progression from its initial to its desired configuration, and also indicates if the robot kept its stability.

Though the control law is defined in the image space, for simulation purpose, we will describe the tasks in $SE(3)$. Defining, for instance, pose of the target with respect to a fixed inertial frame.

5.3.2 Visual Servoing based Positioning Tasks

The positioning task will be considered as completed if the humanoid, using visual feedback, succeeds to walk autonomously from its initial configuration towards a prescribed configuration relative to the target and stop at that desired configuration.

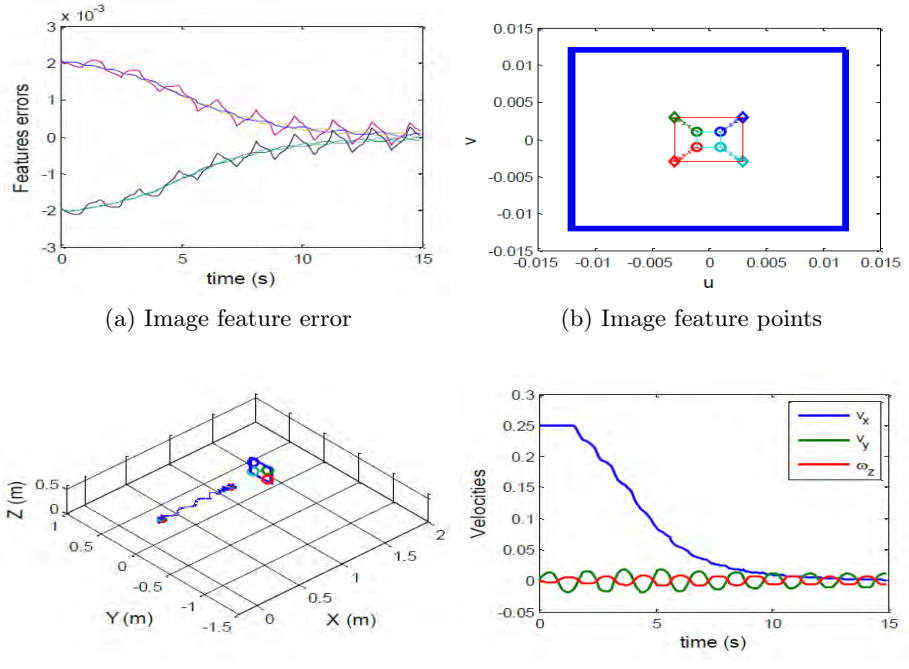
This simulation will focus on translations, rotations and their combinations, since any positioning task, however complex it could be, can be reduced to simple translations, rotations or both.

5.3.2.1 Translations

Longitudinal Translation. Consider a case where the robot has to position itself with respect to a target object located in front of it at $(1.20,\ 0.0,\ 0.465)\,m$ and $(0.0,\ 0.0,\ 0.0)\,rad$. The desired pose of the robot's camera with respect to the target is $(-0.40,\ 0.0,\ 0.0)\ m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$.

The results of this simulation are grouped in Figure 5.4, where all plots respectively, of features error, image feature points, camera and velocities indicate the convergence of the task. In image space (Figure 5.4b), for instance, the feature points converge from their initial configuration (circles with a cyan frame) to their desired configuration (diamonds with a red frame) almost in straight line. Similarly in $3D$ space, the camera goes straight towards the target (Figure 5.4c). The observed oscillations are due to the sway motion induced by the walking. This motion necessary for a stable gait should not be suppressed, but it effects on features can mitigated [19].

Finally, the displacement of the humanoid is shown in Figure 5.5, where can be seen the robot's footsteps from the initial to the desired pose, as well as the reference ZMP (blue) and the actual ZMP (red). These trajectories show not only how the robot walked but also if it kept its stability during execution of the task.



(a) Image feature error

(b) Image feature points

(c) Camera trajectory

(d) Robot's base input velocities

Figure 5.4: Positioning task achieved by simple translation along the X axis

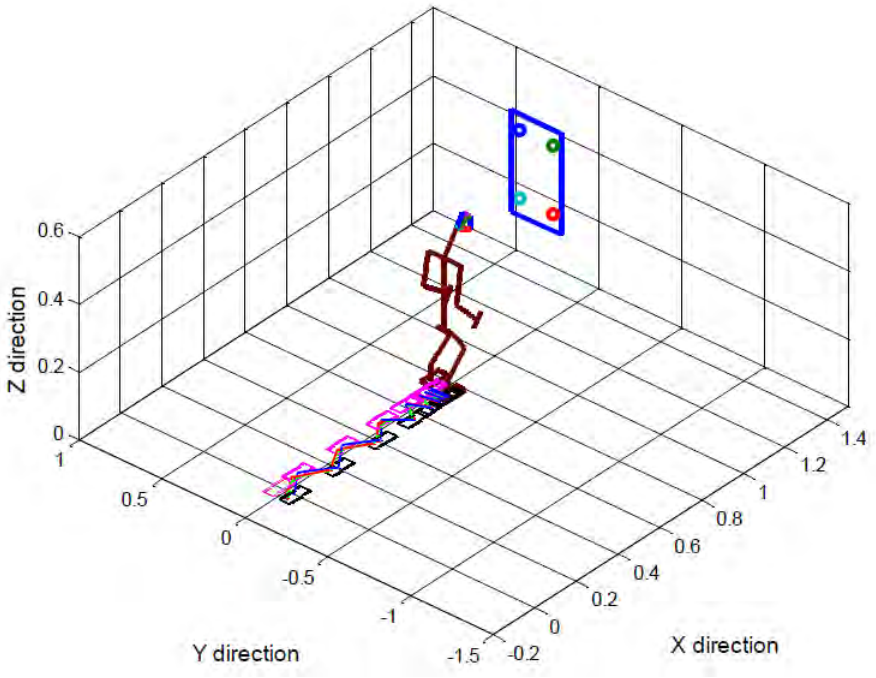


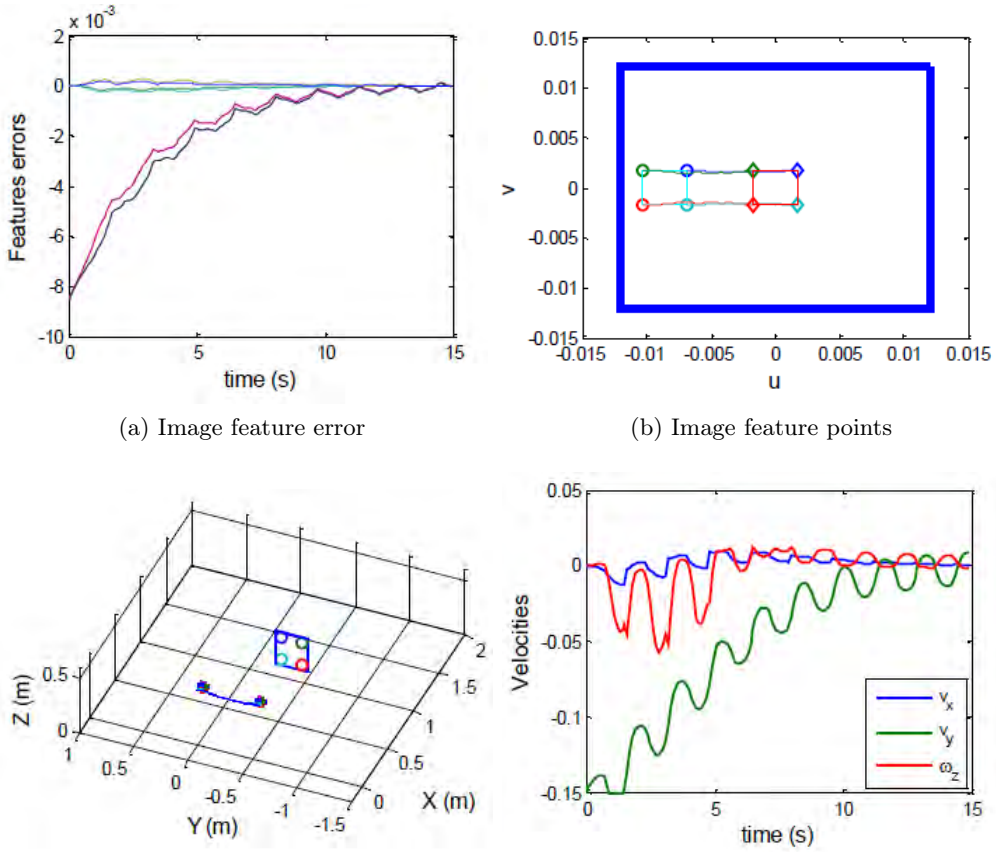
Figure 5.5: Trajectory of the robot during a positioning task along the X axis

Lateral Translation. Consider now a positioning task requiring from the robot a simple translation in Y direction. The target pose is $(0.758,\ -0.50,\ 0.465)\,m$ and $(0.0,\ 0.0,\ 0.0)\,rad$, while the desired camera pose relative to the target is given by $(-0.70,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$. The target pose accounts for the X offset of the camera $(0.058\,m)$ to cancel any translation in the X direction.

In Figure 5.6 are grouped the results of this simulation. Clearly the task converges as indicated in Figure 5.6a, where the horizontal components of the error decrease exponential to zero. Similarly, Figure 5.6b confirms that convergence with good trajectories of feature points in the image.

In Figure 5.6c on the other hand, it can be noticed a slight curvature of the 3D camera trajectory. As shown in Figure 5.6d, this is due to the rotation velocity of the robot trying to point the camera towards the target. It can also be seen how the sway motion affects the input velocities.

Finally, the global motion of the humanoid robot is illustrated on Figure 5.7 where the footsteps and the slight curvature they describe are clearly visible.



(a) Image feature error

(b) Image feature points

(c) Camera trajectory

(d) Robot's base input velocities

Figure 5.6: Positioning task achieved by translation along the Y axis

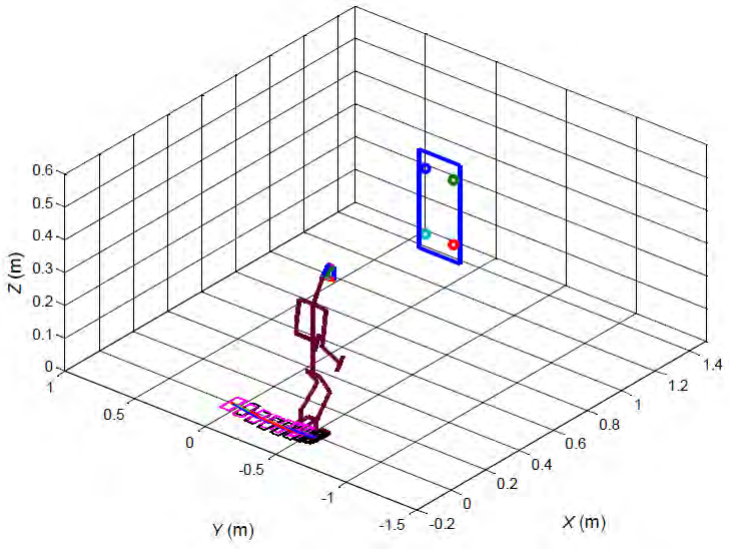
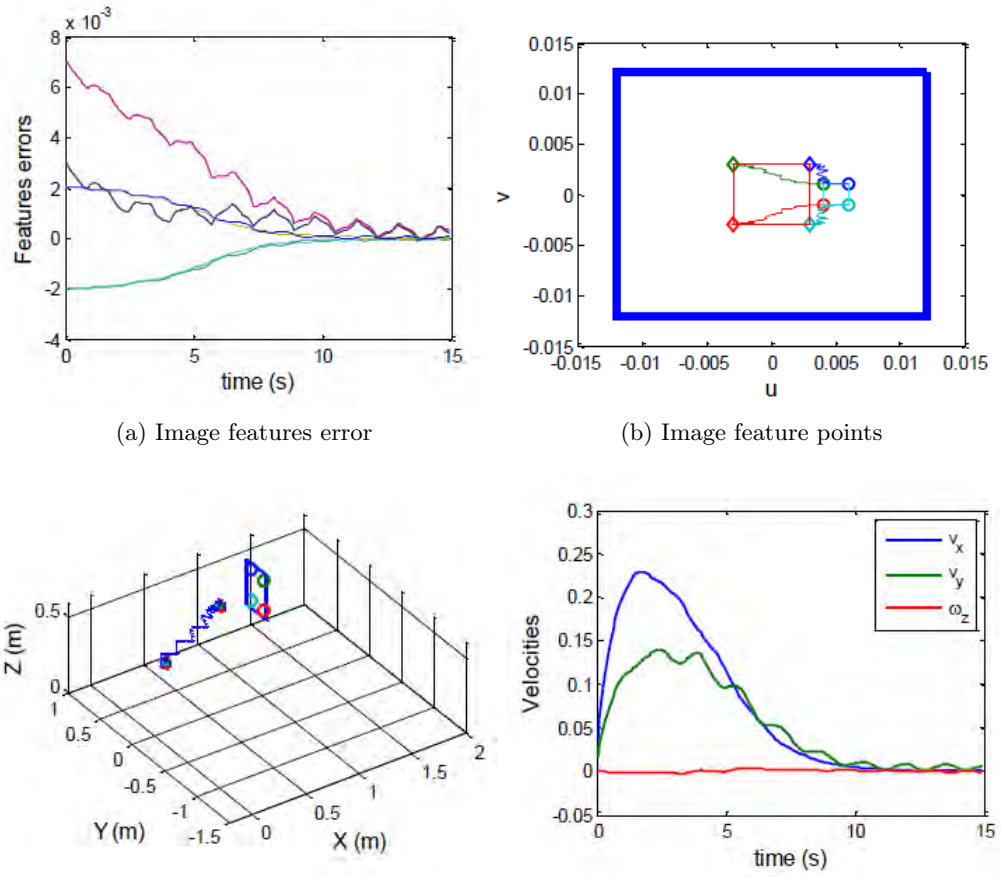


Figure 5.7: Trajectory of the robot during a positioning task along the Y axis

Longitudinal and Lateral Translations. Let us consider now a positioning task requiring X and Y translations. The target object, this time, is located at $(1.258,\ 0.50,\ 0.465)\,m$ and $(0.0,\ 0.0,\ 0.0)\,rad$, and the desired pose of the camera relative to the target is $(-0.40,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$.

The results are grouped in Figure 5.8, where without any doubt the convergence of the task can be noticed. The trajectories of image feature points and that of the camera are satisfactory. With low-pass filters whose cutoff frequencies equal $2.0\,rad/s$, the inputs velocities have been filtered as can be noticed in Figure 5.8d, where the rotational velocity stays almost zero as expected.



(a) Image features error

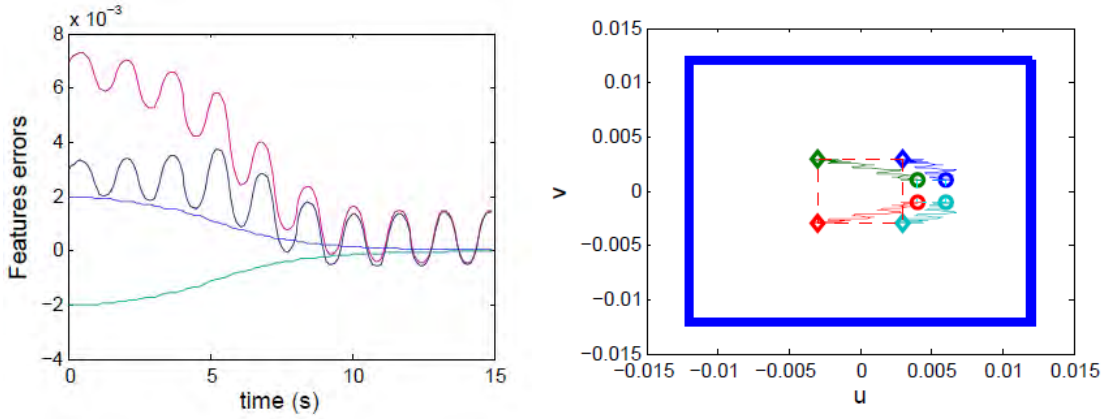
(b) Image feature points

(c) Camera trajectory

(d) Filtered robot's base input velocities

Figure 5.8: Positioning task achieved by combined X and Y translations

At this point, it is worth highlighting the benefit provided by the mechanical compensation scheme (recomputing the neck's joints. Ref. Equation 4.53). Indeed, the attenuation of oscillations affecting the images features can be noticed when comparing Figures 5.8a and 5.8b to Figure 5.9 showing the features behavior without compensation.



(a) Image feature error

(b) Image feature points

Figure 5.9: Features behavior without mechanical compensation of sway motion

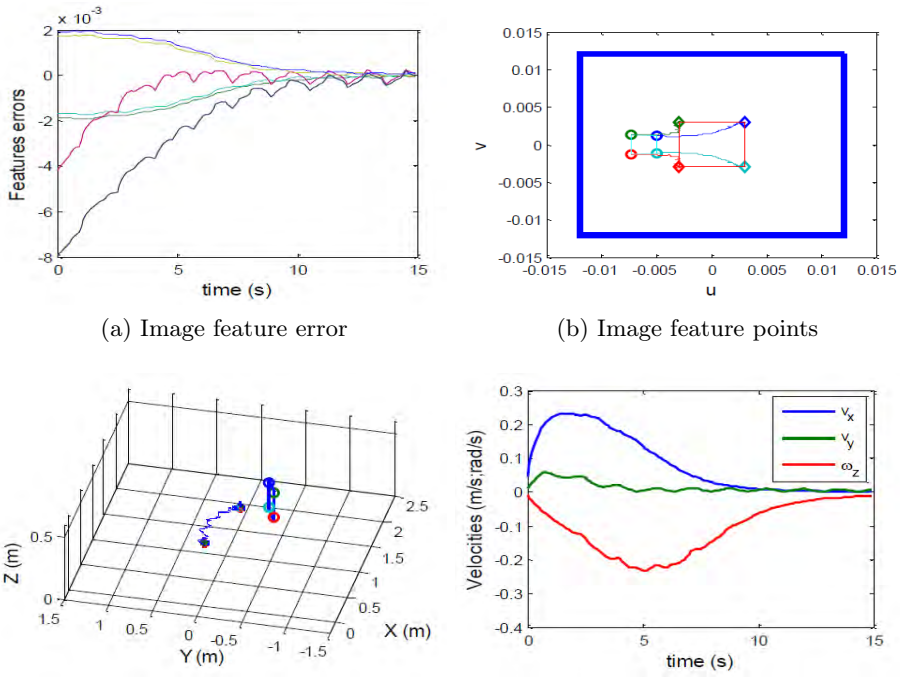
Far from the target, these oscillations almost disappear, and at the desired pose, their magnitude is about 80 % less. Consequently, the positioning accuracy is significantly improved. However, this solution tends to impose high accelerations on the neck joints.

5.3.2.2 Translations and Rotation

The last positioning simulation consists of combining translational and rotational motions. The target pose is $(1.058,\ -0.50,\ 0.465)\,m$ and $(0.0,\ 0.0,\ -\frac{\pi}{3})\ rad$. The desired pose of the robot's camera with respect to the target object is $(-0.40,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$.

The results of this task are shown in Figure 5.10. The task function once again converges to zero as confirmed by the image features error (Figure 5.10a) and the trajectory of image feature points (Figure 5.10b). The camera trajectory is still satisfactory.

The input velocities of the base in Figure 5.10d, analyzed in conjunction with Figure 5.10a, reveals that the observed rotation velocity ω_z after the settling of v_x and v_y is mainly due to an auxiliary task: the humanoid's sight direction and body alignment, which is performed as an internal motion not affecting the main task.



(a) Image feature error

(b) Image feature points

(c) Camera trajectory

(d) Robot's base input velocities

Figure 5.10: Positioning task with combined translations and rotation

5.3.3 Visual Servoing based Tracking Tasks

This section is an application of the theoretical formulation done in Section 4.4. Unlike positioning where the target is fixed and the robot stops at the desired relative pose, in tracking, the target is moving and the robot is required to reach and maintain the relative desired pose while walking. As argued in that Section, we remind the reader that the target motion (velocity ${}^{c}V_{o}$) from the tracking camera/robot perspective, represents the reference trajectory whereas from the task function perspective, whose desired value is always zero, it is seen as a disturbance to be rejected.

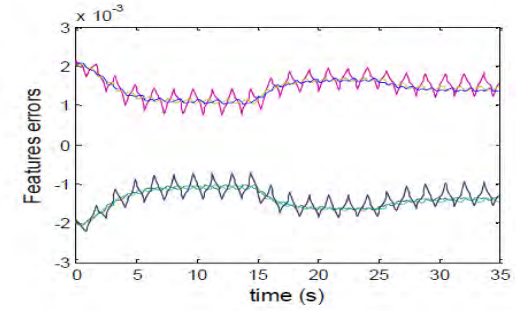
For simulation purposes, the target's velocity with respect to a fixed inertial frame is assumed to be known and it is used in feed-forward (Equation 4.56) to compensated the target motion in the image. Also, the velocity commands will be filtered as previously with a low pass filter ($\omega_c = 2.0\, rad/s$).

Similarly to positioning, the validation will be based on translational motions, rotational motions, and their combinations in order to realize complex tracking tasks. We will also distinguish non-accelerating from accelerating target motions.

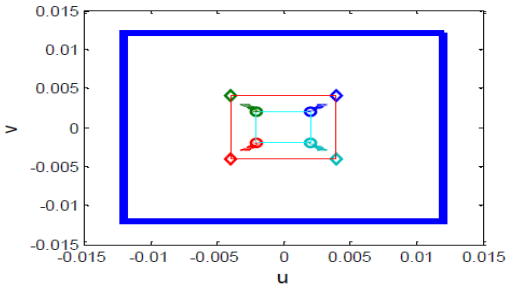
5.3.3.1 Non-Accelerating Target

Longitudinal Target Tracking. Consider a tracking task where the target moves in the X direction with an initial velocity of $0.04\,m/s$ for 15 seconds. Then the velocity is stepped up to $0.08\,m/s$ and finally 10 seconds later, it is stepped down to $0.06\,m/s$. The initial target pose is $(0.658,\ 0.00,\ 0.465)\,m$ and $(0.0,\ 0.0,\ -\frac{4\pi}{15})\,rad$. The desired pose of the robot's camera relative to the target object is $(-0.30,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$.

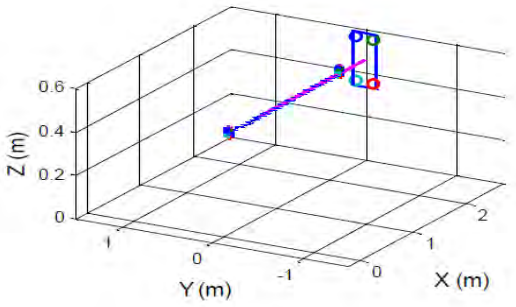
Using Equation (4.49), that is without compensating the target motion, the results of this simulation are shown in Figure 5.11. It can be observed that the robot manages to track the target (Figures 5.11c and 5.11d) but fails to reach the desired relative pose (Figures 5.11a and 5.11b).



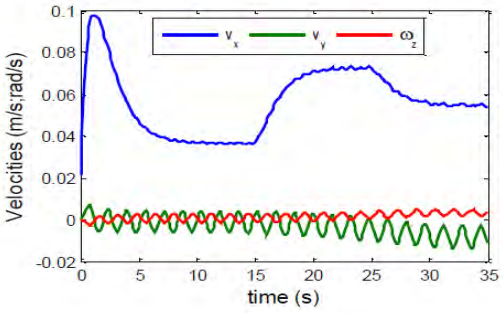
(a) Image feature error



(b) Image feature points



(c) Camera and target trajectories



(d) Robot's base input velocities

Figure 5.11: Tracking a target moving along the X axis

To ease the interpretation of these results, let us rewrite the corresponding camera velocity and closed loop equations (exponential decrease of the task function $\boldsymbol{e}$) as follows

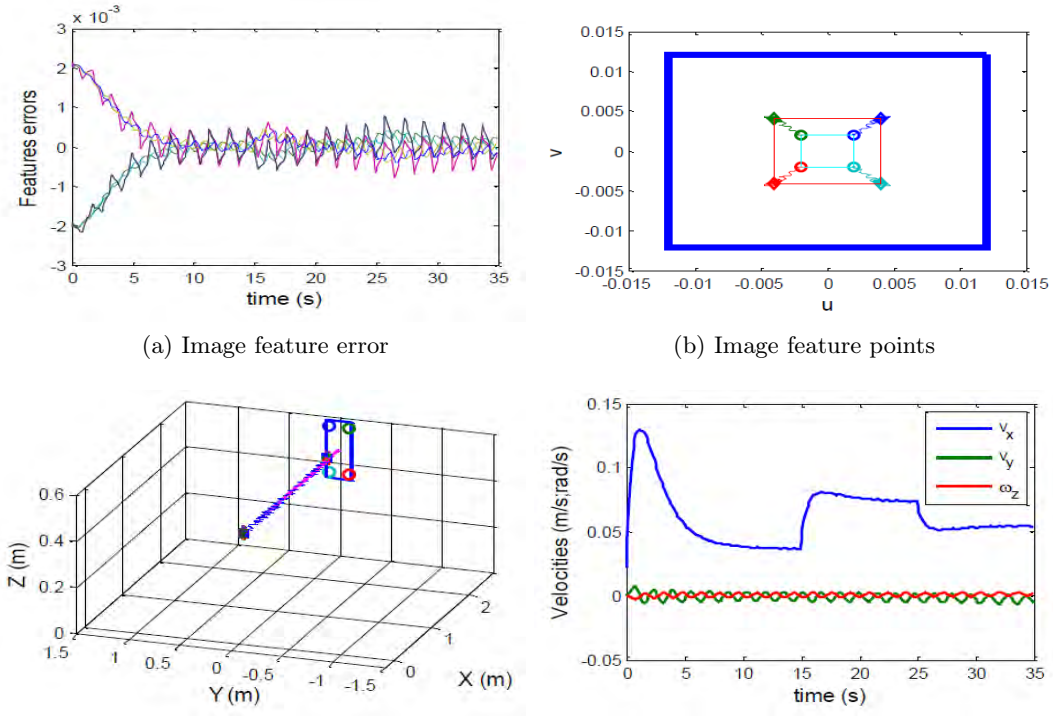
$$\begin{cases} {}^{c}V = -\widehat{\mathbf{L}_e^{\dagger}}\,\Lambda_p \boldsymbol{e} \\ \dot{\boldsymbol{e}} + \mathbf{L}_e \widehat{\mathbf{L}_e^{\dagger}}\,\Lambda_p \boldsymbol{e} = -\mathbf{L}_e\,{}^{c}V_o \end{cases} \tag{5.83}$$

With $\mathbf{L}_e\widehat{\mathbf{L}_e^{\dagger}}$ positive-definite, Equation (5.83) shows that the equilibrium point of the task function $\boldsymbol{e}$ is shifted by the target velocity. This justifies the failure to reach the desire relative pose.

However, when the system settles ($\dot{\boldsymbol{e}} = 0$), the camera velocity becomes now equivalent to that of the target. This can be observed in Figure 5.11d, where v_x after starting with a high value to bring the robot close to the desired relative pose, settles successively at about $0.04\,m/s$, then $0.08\,m/s$ and finally $0.06\,m/s$, which represent respectively the three reference velocities of the target during this task. The observed first order profile of the velocity as per Equation (5.83) stems from that of the task function $\boldsymbol{e}$ (exponential decrease).

On the other hand, v_y and ω_z oscillating in opposition of phase, have started drifting from the $15th$ second which corresponds to the time where the target velocity was first changed. This suggests that the robot reacted to the disturbance (sway motion combined with the step of the target velocity) on $\boldsymbol{e}$ by a slight anticlockwise rotation and now progresses with combined x and y velocities without affecting the direction of the camera. This is confirmed in Figure 5.11c showing 3D straight line trajectories of the camera (blue) and of the target (magenta) along X direction.

After compensating for the target motion (using Equation 4.70), the camera velocity is now ${}^cV = -\widehat{\mathbf{L}_e^{\dagger}}(\Lambda_p \boldsymbol{e} + \widehat{\mathbf{L}_e}{}^cV_o)$. It can now change as soon as the target velocity changes allowing as such to reject the disturbance on $\boldsymbol{e}$, which now converges to zero as shown in Figure 5.12. It also allows to track the target when $\boldsymbol{e}$ settles. The observed oscillations are now about zero and the image feature points (Figure 5.12b) reached and maintain their desired positions.



(a) Image feature error

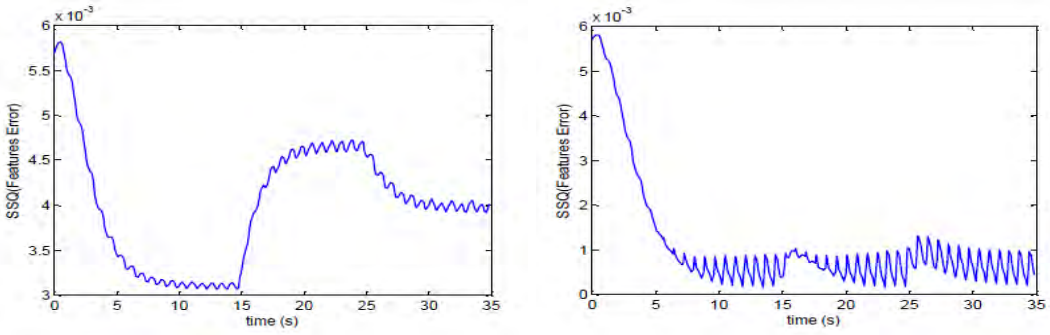
(b) Image feature points

(c) Camera and target trajectories

(d) Robot's base input velocities

Figure 5.12: Tracking with feed-forward compensation of a target moving along the X axis

In Figure 5.12d, the dynamics filter-visual controller is perceptible at each time the target velocity was stepped. The norm of $\boldsymbol{e}$ before and after compensating for the target motion are shown in Figure 5.13.

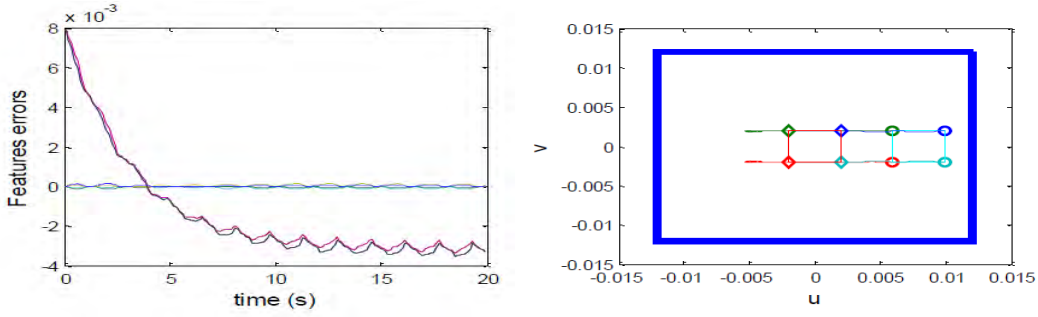


(a) Task function without compensation (b) Task function with compensation

Figure 5.13: Norm of task function without and with target motion compensation

Lateral Target Tracking. Consider now a tracking task where the target, initially located at $(0.658,\ -0.30,\ 0.465)\,m$ and $(0.0,\ 0.0,\ 0.0)\,rad$, moves laterally with a velocity of $-0.05\,m/s$. The desired pose of the robot with respect to the target is $(-0.60,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$. In this task, the robot has to move first with a positive velocity to meet the target and thereafter adopt the negative target's velocity to track it.

Figure 5.14 illustrates how the robot failed to successfully complete such a task without target motion compensation. Some image features went beyond their desired values and were enable to come back.



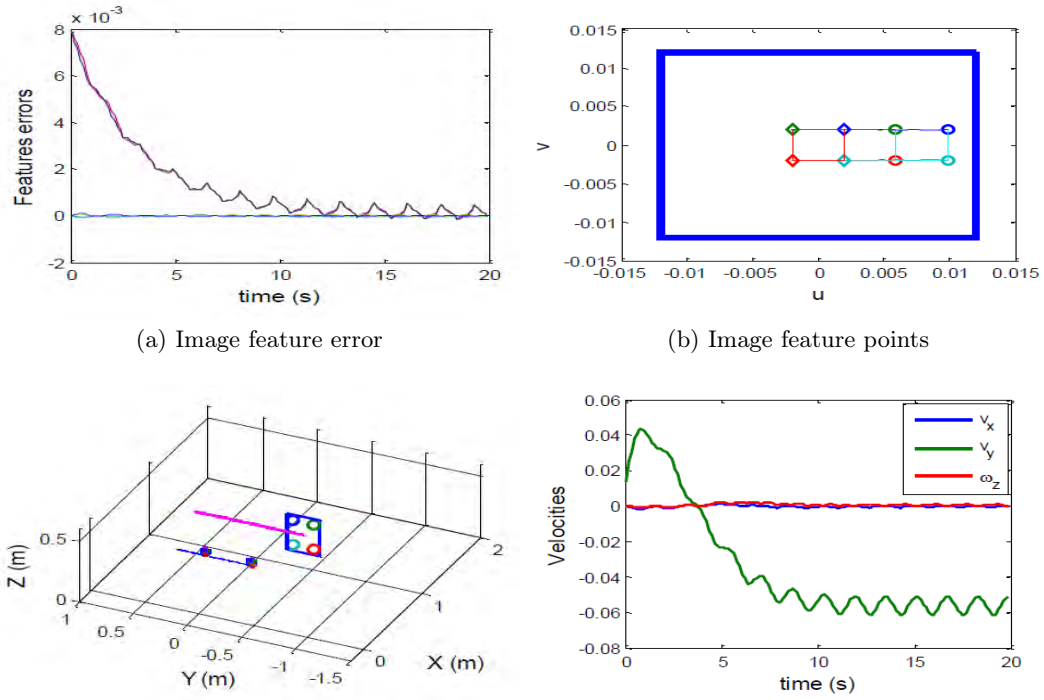
(a) Image features error (b) Image feature points

Figure 5.14: Visual target tracking with lateral velocity and no motion compensation

This observation is justified, as previously, by the shift of the equilibrium point of $\boldsymbol{e}$. How this reflects on the features can be seen by rewriting the closed loop equation explicitly as a function

of the features, hence $\dot{\boldsymbol{\xi}} + \mathbf{L}_e\widehat{\mathbf{L}_e^{\dagger}}\,\Lambda_p\boldsymbol{\xi} = \mathbf{L}_e\widehat{\mathbf{L}_e^{\dagger}}\Lambda_p\boldsymbol{\xi}^d - \mathbf{L}_e{}^cV_o$, ($\dot{\boldsymbol{\xi}}^d = 0$). Clearly $\boldsymbol{\xi}$ will not settles at $\boldsymbol{\xi}^d$, since $-\mathbf{L}_e{}^cV_o$ is similar to an input disturbance acting directly on the set-point.

This problem is corrected after compensation as shown in Figure 5.15. The "rendezvous" target-robot is successful. Thanks to the feed-forward action, the robot anticipatively changed the direction of its velocity (at time $= 3\,s$ in Figure 5.15d) to allow a decrease of the task function with no overshoot of the images features (see Figure 5.15a and Figure 5.15b).



(a) Image feature error

(b) Image feature points

(c) Camera and target trajectories

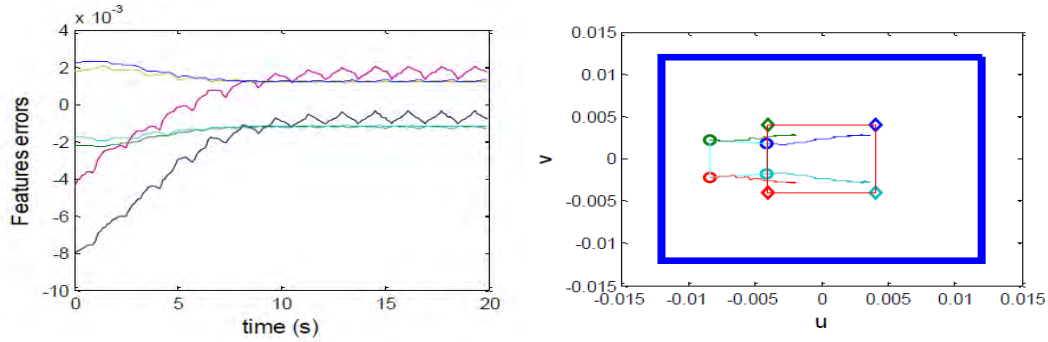
(d) Robot's base input velocities

Figure 5.15: Visual target tracking with lateral velocity and motion compensation

Longitudinal and Lateral Target Tracking. Consider now a task consisting of reaching and maintaining a pose of $(-0.30,\, 0.0,\, 0.0)\,m$ and $(\frac{\pi}{2},\, 0.0,\, \frac{\pi}{2})\,rad$ of the robot's camera with respect to a target, whose initial pose is $(0.658,\, -0.30,\, 0.465)\,m$ and $(0.0,\, 0.0,\, -\frac{\pi}{4})\,rad$ and which moves at $0.04\,m/s$ and $-0.03\,m/s$ in X and Y directions, respectively.

Once again, Figure 5.16 tells us that, without compensation of the target motion, the robot manages to follow the target (constant final value of features error) but fails to reach the relative desired configuration. The reason of this failure is the same as in the two previous cases.

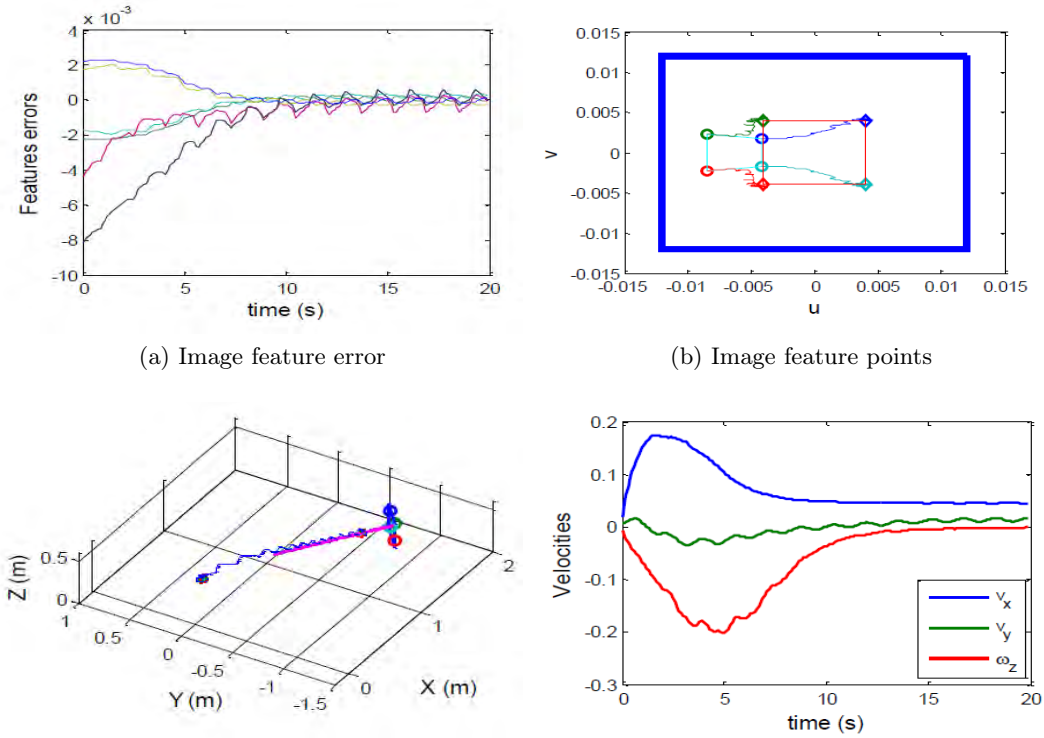
As shown in Figure 5.17, implementing the target compensation scheme corrects this problem and enables the robot to reach its desired pose while tracking the target.



(a) Image feature error

(b) Image feature points

Figure 5.16: Tracking a target moving with X and Y velocities



(a) Image feature error

(b) Image feature points

(c) Trajectories of camera (blue) and target (magenta)

(d) Robot's base input velocities

Figure 5.17: Tracking a target moving with X and Y velocities

5.3.3.2 Accelerating Target

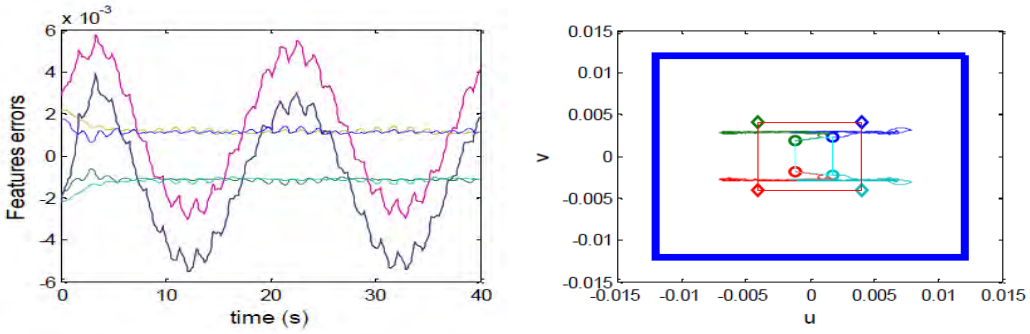
Up to now, in all tracking applications that were considered, the target's motion was characterized by zero acceleration. Let us, in the sequel, consider two applications with accelerating targets.

Target with Sinusoidal Motion. Consider a tracking task where the target describes a sinusoidal motion along an axis orientated at $\frac{\pi}{4}$ with respect to the X axis. The X and Y velocities of the target are given by

$$\begin{aligned} \dot{x}_{target} &= a\cos(\frac{\pi}{4}) - \Omega b\cos(\Omega t)\cdot\sin(\frac{\pi}{4}) \\ \dot{y}_{target} &= a\sin(\frac{\pi}{4}) + \Omega b\cos(\Omega t)\cdot\cos(\frac{\pi}{4}) \end{aligned} \tag{5.84}$$

with $a = 0.04\,m/s$, $b = 0.20\,m/s$ and $\Omega = 0.314\,rad/s$. The initial target pose is $(0.658,\ 0.00,\ 0.465)\,m$ and $(0.0,\ 0.0,\ \frac{\pi}{4})\,rad$ and the desired pose of the robot's camera with respect to the target is $(-0.30,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$.

Grouped in Figure 5.18, the results of this simulation without compensation, show how the target's motion affects drastically the features errors. Some image features oscillate with the same frequency as the target, which means failure in reaching the desired pose.



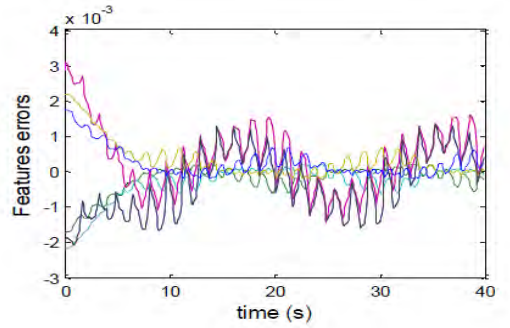
(a) Image features error

(b) Image features trajectories

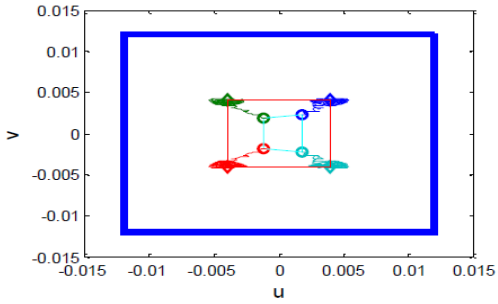
Figure 5.18: Visual tracking of sinusoidal target's motion without feed-forward compensation

As shown in Figure 5.19, when applying the feed-forward compensation, the oscillations magnitude is reduced by 80 % (from $\pm 6\cdot 10^{-3}$ to $\pm 1\cdot 10^{-3}$), the residual error is in part due to the phase-lag between the actual target velocity and its value used in the control action. Another part come from oscillations due to the variation of ω_z which skews the image of the target and thereby affects the horizontal as well as vertical components of the features as can be seen in Figure 5.19a. Despite these small oscillations, Figure 5.19b confirms that the feature points are at their desired values.

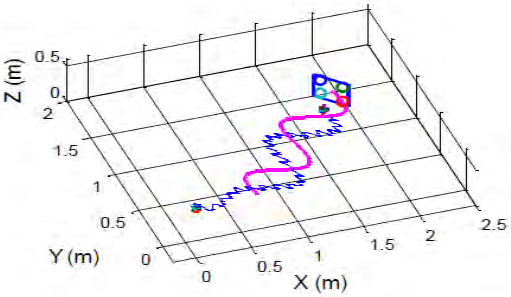
Figure 5.19d tells that the robot, in the first five seconds, moved with a forward motion slightly to its left hand side plus an anticlockwise reorientation in order to face the target, and then switched its lateral velocity to reduce the positioning error ($\boldsymbol{e} \approx 0$). Thereafter, the tracking could start. From the $10th$ second, v_x stabilized at about $0.04\,m/s$, the value of a, while v_y and ω_z have sinusoidal profiles with the target's frequency Ω. Finally, the trajectory of the robot following this complex motion is shown in Figure 5.20.



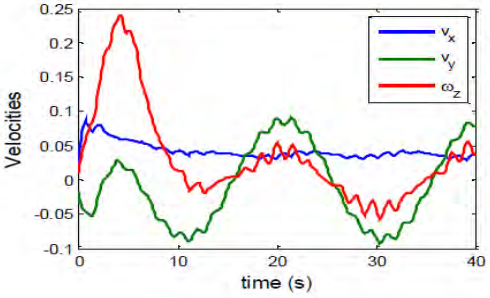
(a) Image feature error



(b) Image feature points



(c) Trajectories of camera (blue) and target (magenta)



(d) Robot's base input velocities

Figure 5.19: Tracking a target moving with X and Y velocities

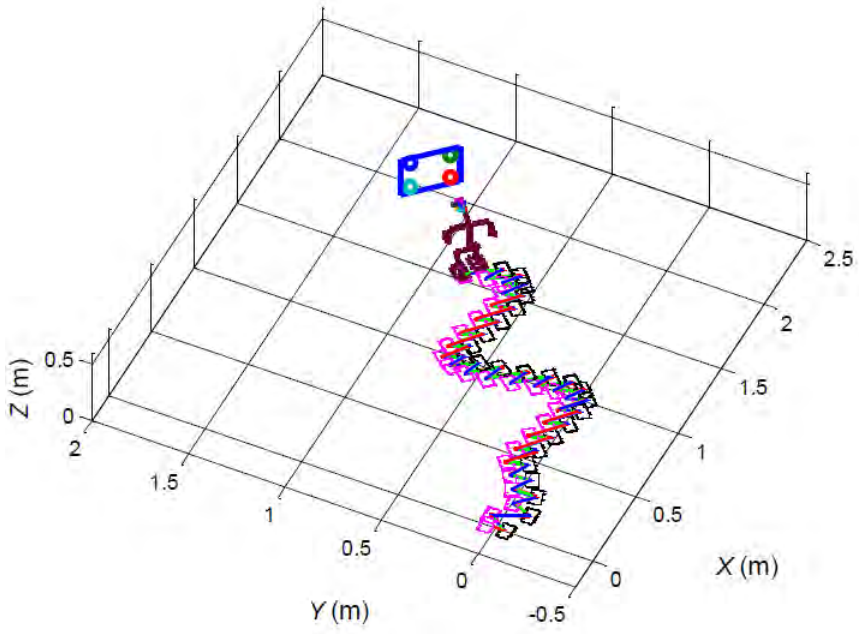


Figure 5.20: Trajectory of the robot following a sinusoidal target's motion

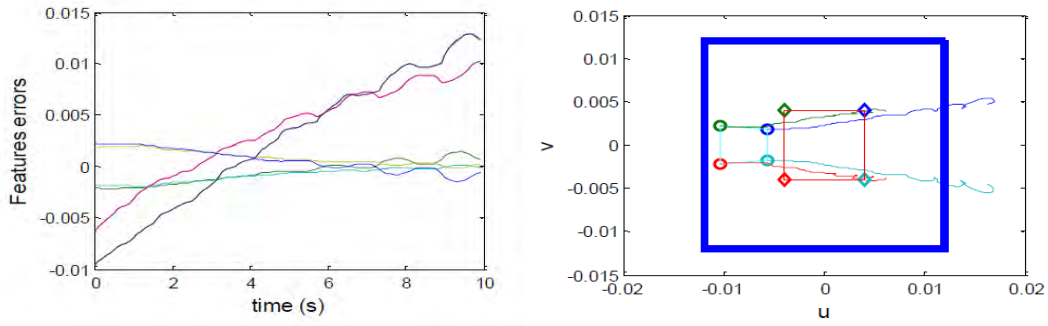
Target with Curvilinear Motion. Consider now a target describing a circular arc with respect to the origin of axis. Initially located at $(0.658,\ -0.40,\ 0.465)\,m$ and $(0.0,\ 0.0,\ -\frac{\pi}{6})\,rad$, the target's frame rotates about its vertical axis with $0.10\,rad/s$ while translating with the following velocities

$$\begin{aligned} \dot{x}_{target} &= -r_t \sin(\varphi - \alpha)\dot{\varphi} \\ \dot{y}_{target} &= \ r_t \cos(\varphi - \alpha)\dot{\varphi} \end{aligned} \tag{5.85}$$

with $r_t = \sqrt{(0.658)^2 + (0.40)^2}$, $\alpha = \arctan\left(\frac{-0.40}{0.658}\right)$ and $\dot{\varphi} = 0.10\,rad/s$.

The desired pose of the camera relative to the robot is $(-0.30,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$. It is chosen such that the robot walks towards the target and starts translating laterally while continuously changing its orientation.

Without any target motion compensation, the features errors and the image feature trajectories shown in Figure 5.21 indicate that the robot is unable to follow such a motion. The feature points leave the camera field of view (see Figure 5.21b) and therefore the servoing stops.



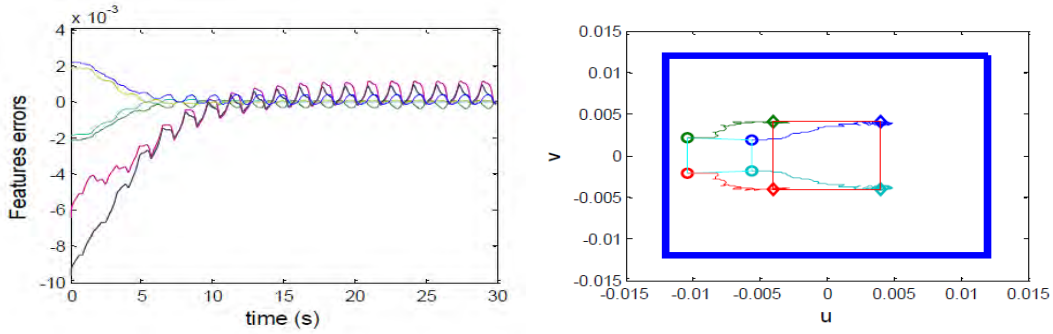
(a) Image feature error

(b) Filtered robot's base input velocities

Figure 5.21: Image features errors and trajectories while tracking a curvilinear target's motion

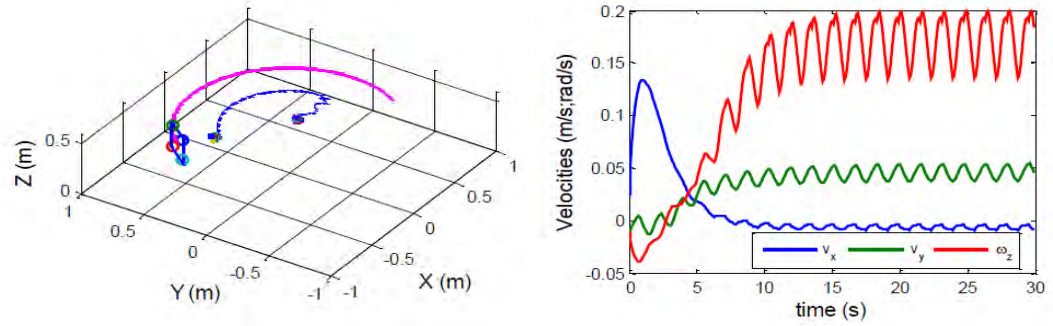
Applying the feed-forward compensation solves this problem as shown in Figure 5.22. Although a small positioning error is perceptible, the task globally converges, the robot tracks this circular trajectory as can be seen in Figure 5.22c.

Once the robot at the relative desired pose, v_x decreases and settles to zero, while v_y and ω_z on average settle respectively at $0.05\,m/s$ and $0.17\,rad/s$ (see Figure 5.22d). These constant lateral translations combined with constant reorientations yield the circular trajectory of the robot shown in Figure 5.23 and thereby contribute to the success of the task.



(a) Image feature error

(b) Image feature points



(c) Trajectories of camera (blue) and target (magenta)

(d) Robot's base input velocities

Figure 5.22: Tracking task of target with curvilinear motion

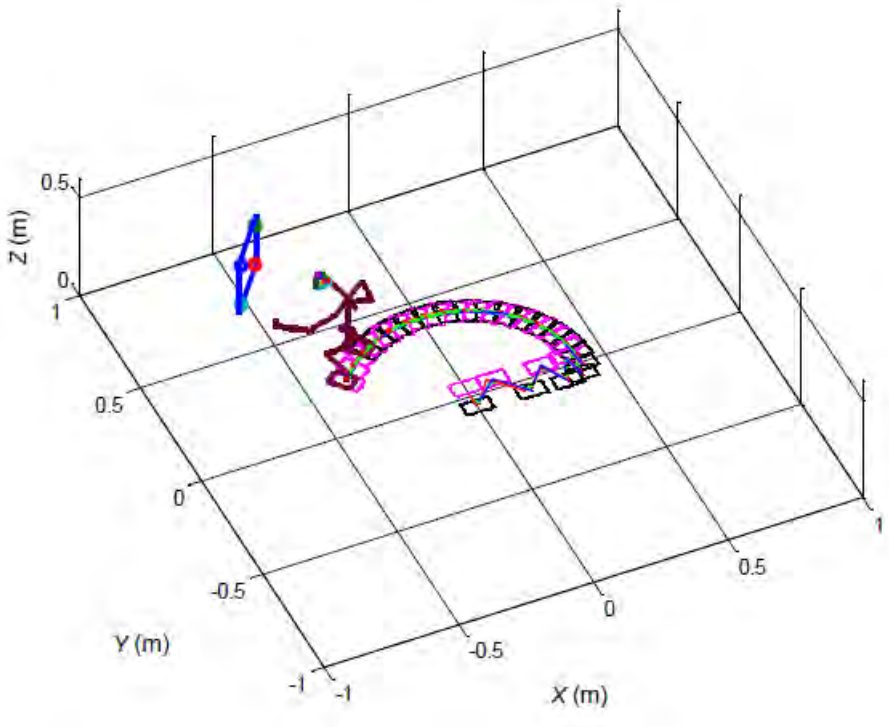


Figure 5.23: Trajectory of the robot tracking a circular target motion

5.3.4 Grasping

Consider now a task consisting of grasping a small moving object while walking. The robot has to walk towards a given target, when close enough, stretch its hand and grasp it. In this simulation, two target objects are used. The first object, employed for relative positioning is a square as in the previous simulations. Its goal is to provide visual feedback to bring the robot near a smaller object, here a red star, to be grasped. The two object are assumed to be rigidly linked and close to each other. Both IBVS and PBVS are exploited respectively for positioning or tracking, and for grasping. The grasping task is triggered when the norm of the IBVS task function reaches a certain threshold (0.001 in this simulation). The PBVS gain is set to 0.5.

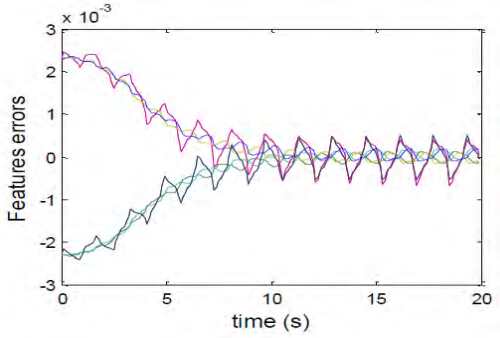
Both targets translate at $0.04\,m/s$ in X direction and $0.02\,m/s$ in Y direction. The initial pose of the square target is $(0.658,\ 0.00,\ 0.465)\,m$ and $(0.0,\ 0.0,\ 0.0)\,rad$ and the relative desired pose of the robot's camera is $(-0.30,\ 0.0,\ 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$. Defined with respect to the square target, the pose of the star target is $(-0.14,\ 0.12,\ -0.12)\,m$ and $(-\frac{\pi}{2},\ 0.0,\ 0.0)\,rad$.

The results of this experiment are shown in Figure 5.24, where can be seen the features errors related to the tracking task (Figure 5.24a), its corresponding norm with indication of the threshold activating the grasping task (Figure 5.24b) and the error on the grasping pose (Figure 5.24c).

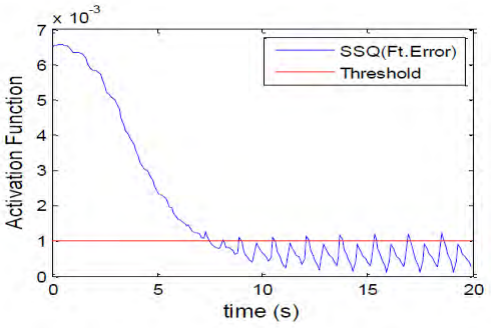
It can be noticed that for about 7.5 seconds, all components of the grasping pose are constant except Δx, which corresponds to the forward direction and decreases as the robot nears the target. Once the grasping task is triggered, all components converge exponentially towards zero. The observed oscillation is due to the sway motion affecting the z axis of the left hand end-effector. Also, the constant error observed on $\theta\mu_z$ can be explained by the fact that the desired hand pose imposes six constraints on the end-effector configuration while the robot's hand has only five DoFs to realize it.

A way to manage this problem could be to define priority between degrees of freedom. In our case for example the hand's orientation has priority over its position.

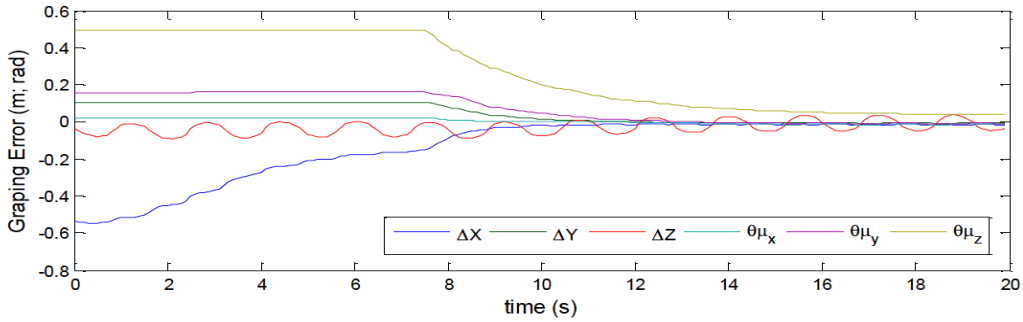
The velocities of the robot's base and the hand joints while performing this task are grouped in Figure 5.25. The trajectory of the robot and the hand trajectory as the robot tracks and grasps the star object are shown in Figure 5.26.



(a) Image Feature Error

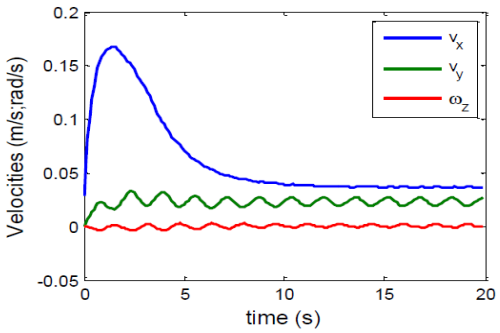


(b) Activation function of the grasping task

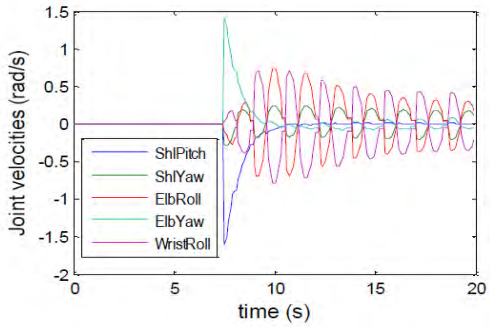


(c) Pose errors of the grasping task

Figure 5.24: Error functions on Visual Servoing based Object Tracking and Grasping



(a) Base's velocities



(b) Hand joints velocities

Figure 5.25: Base and hand velocities during the tracking and grasping tasks

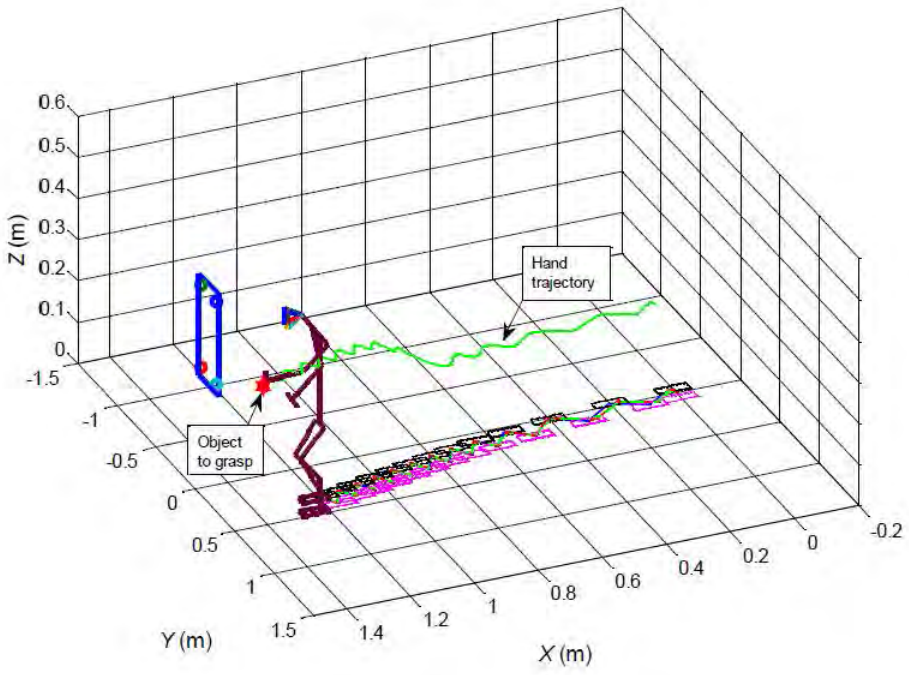


Figure 5.26: Humanoid robot grasping the target at the desired pose

5.4 Experimental Results on Visual Servoing of Humanoid NAO

In this part, the different results obtained by testing our visual servoing algorithms on an actual robot are presented. The experimental platform from the hardware and software points of view has been presented in Section 5.1. Then, we will only give some implementation details and present the experiments with their corresponding results. All related codes and videos can be found in the CD accompanying the thesis.

5.4.1 Implementation Details

All visual servoing applications presented here were implemented in C/C++ and were compiled with NAOqi SDK on a Linux PC. The Open source Computer Vision library (OpenCV) [171] was used for some preliminary experiments such as the camera calibration. More importantly, the Visual Servoing Platform (ViSP) [172] which is a C++ based library developed by the Lagadic group of INRIA/IRISA Rennes, was used for image grabbing and processing, for visual features extraction, modeling and tracking, and also for the definition of task functions and control laws.

Using ViSP library, we have coded in C++ all applications presented in Chapter 4, the model of the robot presented in Chapter 5, especially the forward kinematics and all the Jacobian matrices necessary for the implementation. These codes are given in the attached CD. However, the reactive pattern generator proposed in Chapter 3, apart from simulations, was not implemented on

actual the robot; instead we relied on NAO's own locomotion module which is based on *Wieber*'s formulation [142][167], thus losing a bit of reactivity.

In order to ease the image processing, we kept simple the target by using a white sheet of paper with four black dots spaced by 20 cm and forming a square. The feature points are then the centers of gravity of the dots. The interaction matrix uses the current features with their desired depths. The servoing gain Λ was set to 2.0.

As preliminary experiments the camera was calibrated using a model without distortion. The obtained parameters are $p_x = p_y = 558$, $u_o = 235$ and $v_o = 319$ all in $pixels/m$, which give a re-projection error of 0.36. On the robot side, the veering behavior were compensated by applying the base's velocities as follows

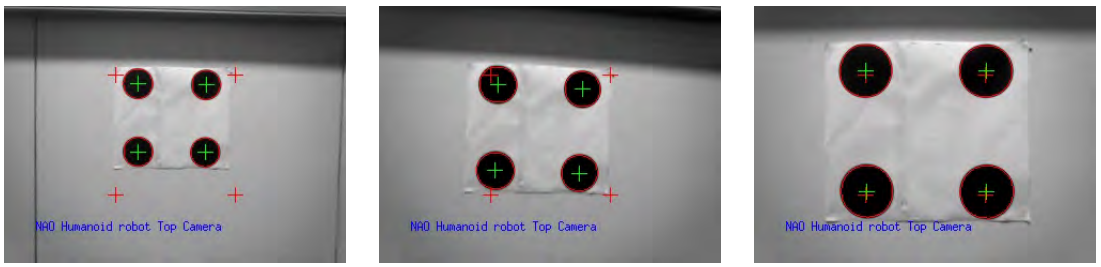
$$\begin{bmatrix} v_{xb} \\ v_{yb} \\ \omega_{zb} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -\rho_{xy} & 1 & 0 \\ -\rho_{x\omega} & -\rho_{y\omega} & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega_z \end{bmatrix} \tag{5.86}$$

with $\rho_{xy} = 0.015$, $\rho_{y\omega} = 0.3$ and $\rho_{x\omega} = \frac{1}{1+e^{-\beta(\|v_x(0)-v_x(t)\|-\rho\|v_x(0)\|)}}$ where $\beta = 850$ and $\rho = 0.15$.

5.4.2 Positioning Experiments

To validate experimentally the visual servoing based positioning scheme, two tasks were considered: a longitudinal translation and combined translations and rotation.

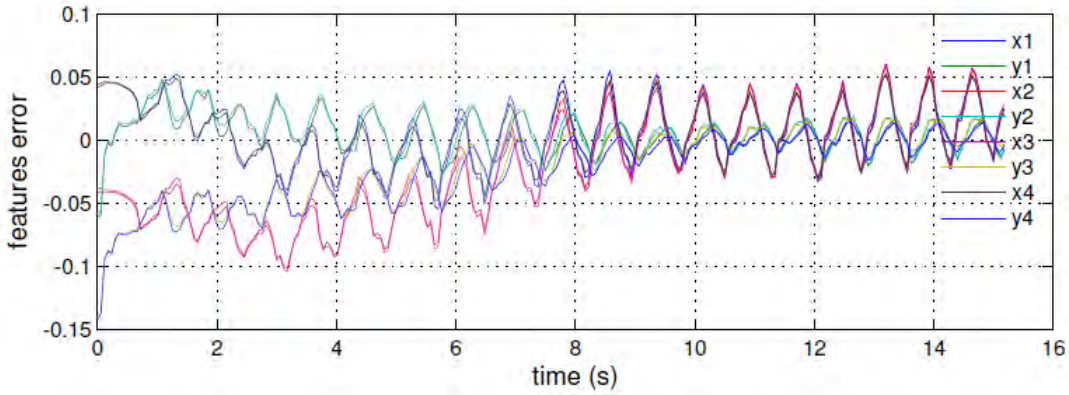
Longitudinal translation. The task being directly specified in the image, we show in Figure 5.27 a sequence of three images related to the task. The observed green crosses on the dots and the red crosses indicate respectively the current and desired image features. Thus, Figure 5.27a shows the initial image configuration, Figure 5.27b an intermediate configuration, while Figure 5.27c shows the configuration after convergence of the task.



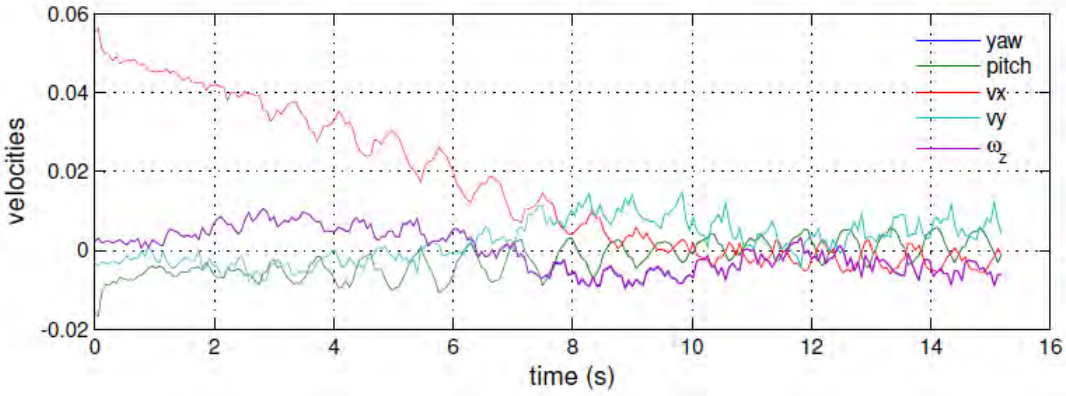
(a) Initial image Configuration (b) Intermediate Configuration (c) Desired image Configuration

Figure 5.27: Image configurations corresponding to positioning task with longitudinal translation

The features error and the reference velocities of neck joints and humanoid's base resulting from this experiment are shown in Figure 5.28. They all indicate the convergence of the task, as confirmed in Figure 5.29 showing the trajectories of the feature points from their initial configuration to their final configuration during this task.



(a) Features Error



(b) Velocity commands

Figure 5.28: Experimental positioning task with longitudinal translation

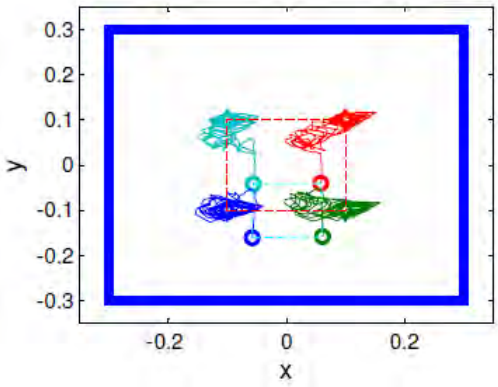
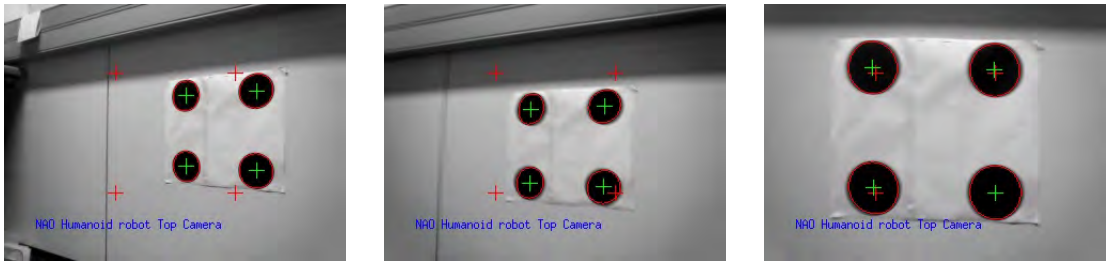


Figure 5.29: Feature trajectories during experimental positioning task

The effects of the sway motion are much more pronounced than in simulation, mainly because the mechanical compensation was disabled due to high accelerations it was requiring. Also, the impact of the feet on the floor coupled with the elasticity of links and joints, the backlash in the gearboxes induced additional oscillations, perceptible through the shakiness of the robot and reflected in the features.

Also, the observable vertical shift at the beginning of the features trajectories (Figure 5.29) is due to the sudden change of the robot's posture from rest when the servoing is initialized to the walking when the robot tends to keep upright its posture.

Combined translations and rotation. Similarly to the previous case, the image specification of this positioning task is illustrated in Figure 5.30, showing a sequence of three images corresponding to three different configurations: initial, intermediate and final.



(a) Initial image Configuration (b) Intermediate image Configuration (c) Desired image Configuration

Figure 5.30: Image configurations corresponding to positioning task with translations and rotation

The features error, the image features trajectories and the robot's input velocities are grouped in Figure 5.32. Once again, the task convergence can be noticed; The robot succeeded in walking from its initial configuration to the desired configuration as prescribed in the image.

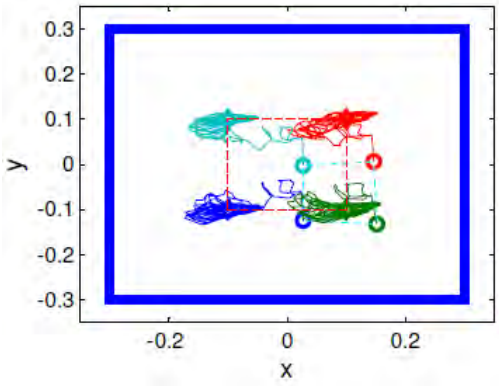
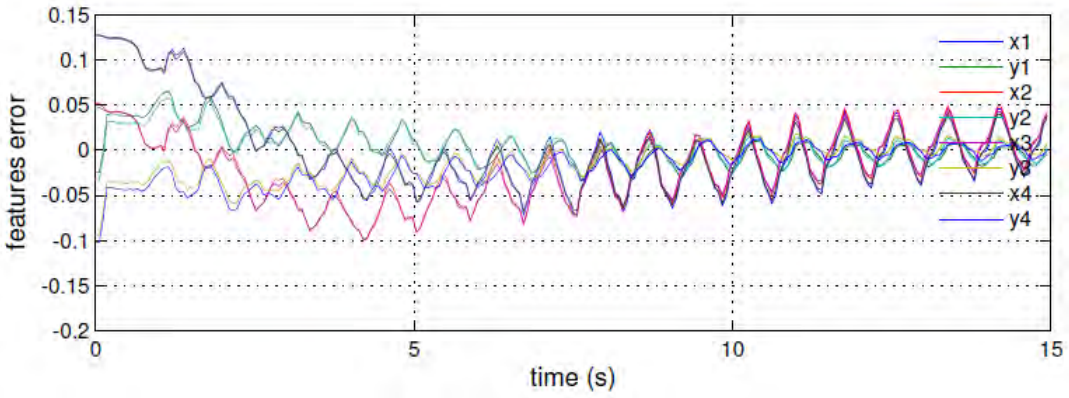
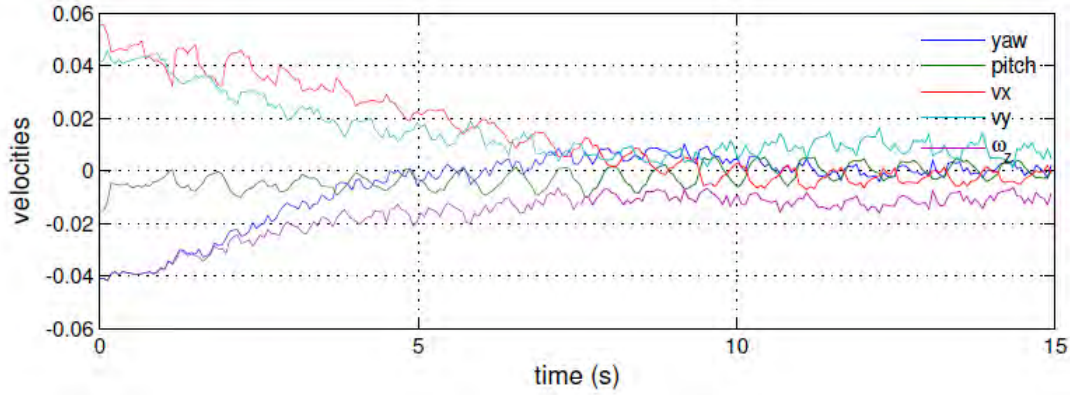


Figure 5.31: Feature trajectories during positioning task with translations and rotation

The general motion of the robot can be confirmed in Figure 5.32b, particularly with the base's velocities. Additionally, an internal motion: the sight's direction and body alignment is perceptible through the non-zero value of ω_z after the task convergence and settling of the yaw velocity to zero.



(a) Feature Error



(b) Velocity Commands

Figure 5.32: Experimental positioning task with translations and rotation

5.4.3 Tracking Experiments

To validate the tracking scheme, two experiments were conducted. As shown in Figure 5.33, the mobile target was realized by setting visual markers on a remote controlled (RC) car.

For compensation purposes, based on the image and robot's kinematic measurements, we configured a Kalman filter with constant acceleration and colored noise model found in ViSP library [173] to estimate the target motion (see Appendix C for more details).

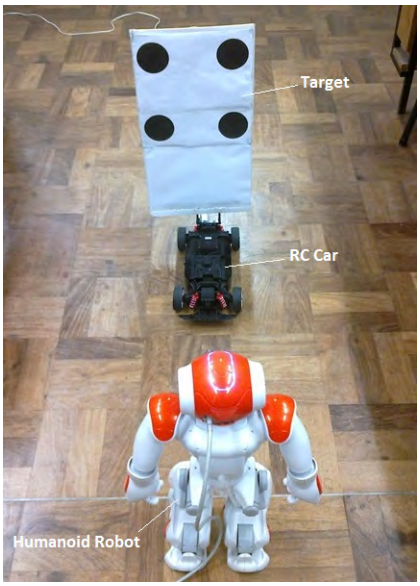


Figure 5.33: Experiment setup of Tracking tasks

The target in the first tracking task translates longitudinally with respect to the robot, while in the second one, the two translations are combined.

Longitudinal Tracking. This experiment was conducted with a slowly moving target and its compensation scheme disabled.

The image features trajectories during this tracking task are shown in Figure 5.34. The observed behavior of the features going at the beginning towards the left before coming back towards their expected trajectories is due to the veering behavior which is particularly significant on our experimental platform.

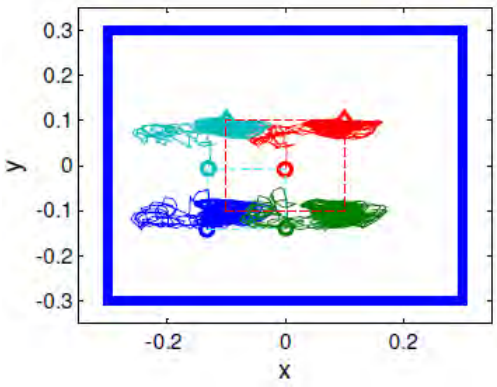
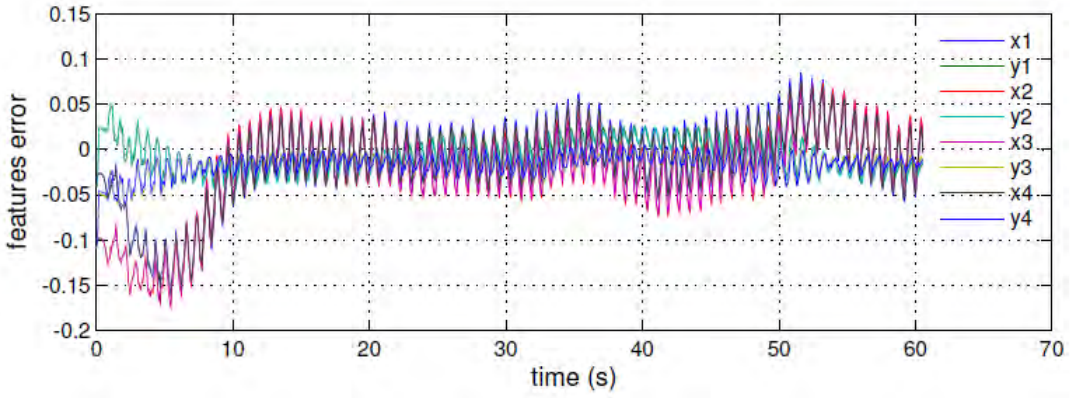
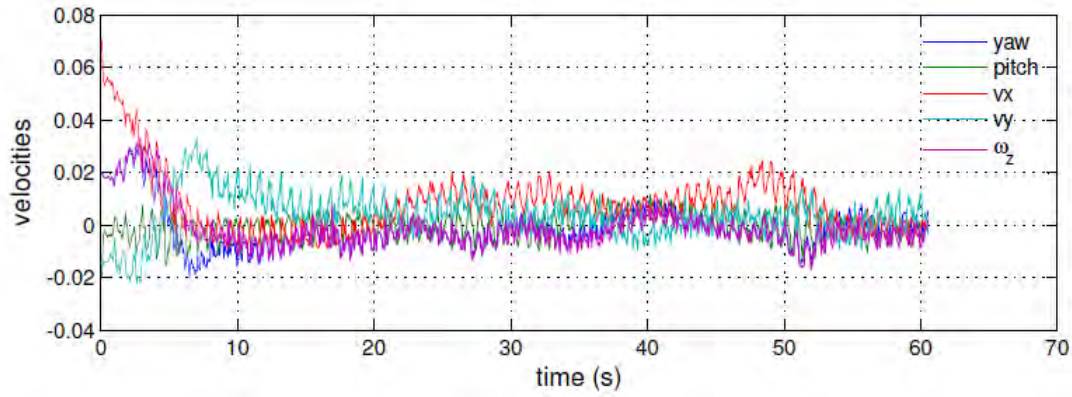


Figure 5.34: Image features trajectories during longitudinal tracking

The features error and the robot's base velocities resulting from this experiment are shown in Figure 5.35. It can be observed that for the first 20 s, the robot walked to reach the relative desired pose, and thereafter the tracking started and ran for about 25 s. The latter could be noticed by the rising, the settling and the decrease of v_x while the remaining velocities stayed about zero.



(a) Features error

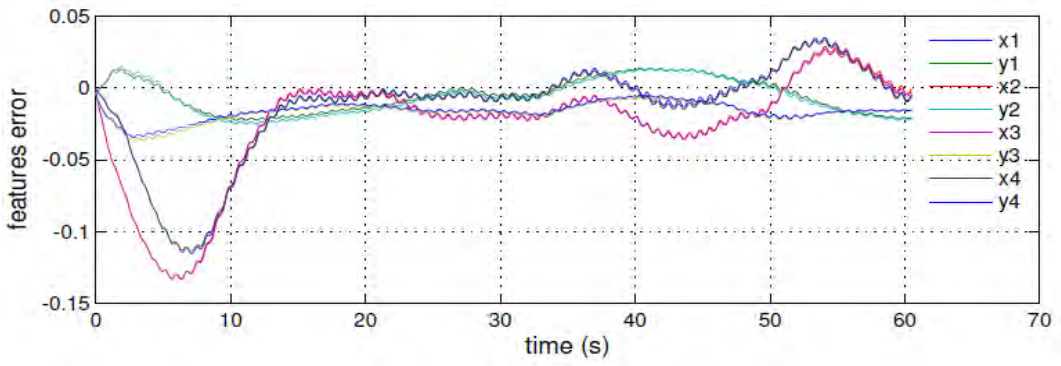


(b) Reference velocities

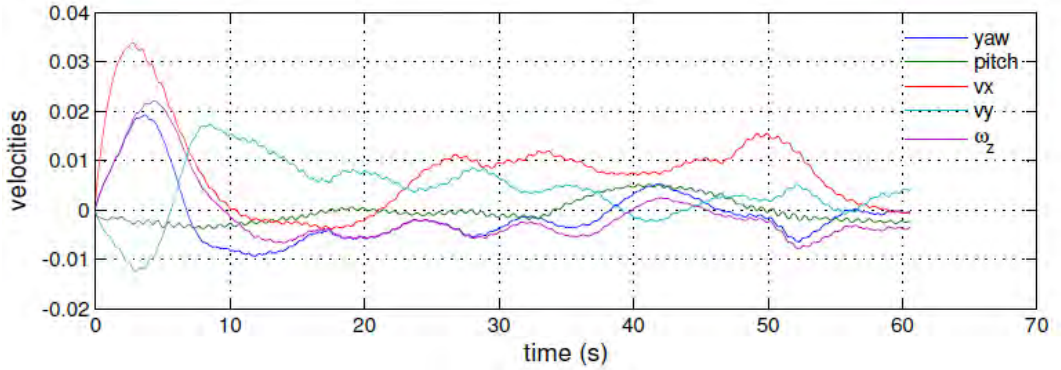
Figure 5.35: Experimental longitudinal tracking task

In Figure 5.36 are shown the filtered features error and velocities which give better insight of the task execution. Similarly to simulation cases without compensation, the positioning error is now more clear in Figure 5.36a. Nevertheless, the robot managed to track the target confirming thus the theory and the simulations.

Finally, a sequence of six images, illustrating different features configurations during the tracking, is shown in Figure 5.37. It can be seen the initial and the desired configuration which was achieved when the target stopped.



(a) Features Error



(b) Reference velocities

Figure 5.36: Filtered signals of the experimental longitudinal tracking task

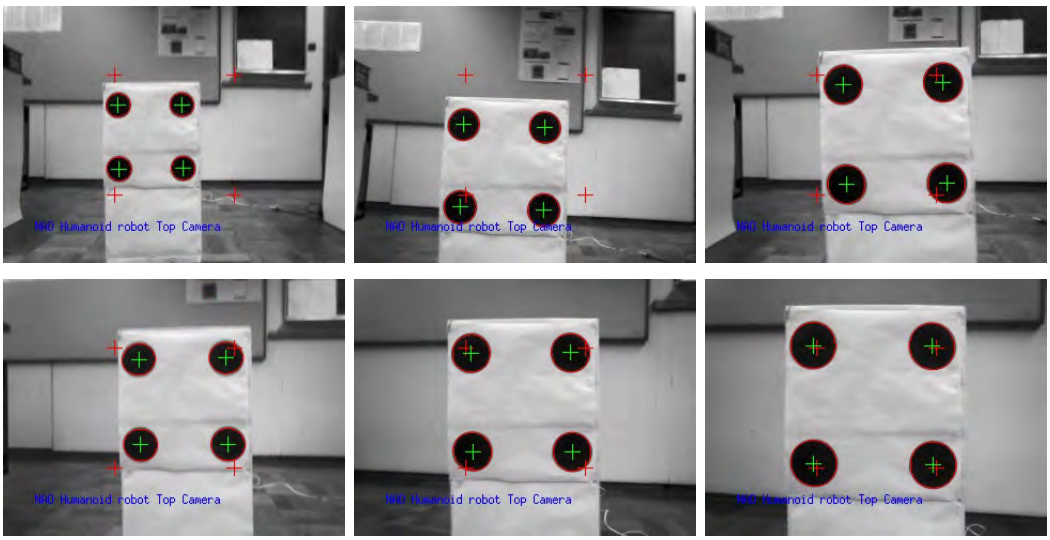


Figure 5.37: Sequence of image configurations during longitudinal tracking experiment

Longitudinal and Lateral Tracking. In this experiment, the compensation scheme is now enabled. The longitudinal and lateral tracking is defined with respect to a fixed inertial frame whose origin of axes is located between the feet of the robot at the initial position and whose X and Y axes are defined, respectively, at the intersections between the robot's sagittal and frontal planes with the ground.
The target is located initially in front of the robot but slightly rotated anticlockwise as shown on the first image in Figure 5.38, illustrating six different features configurations during the task.

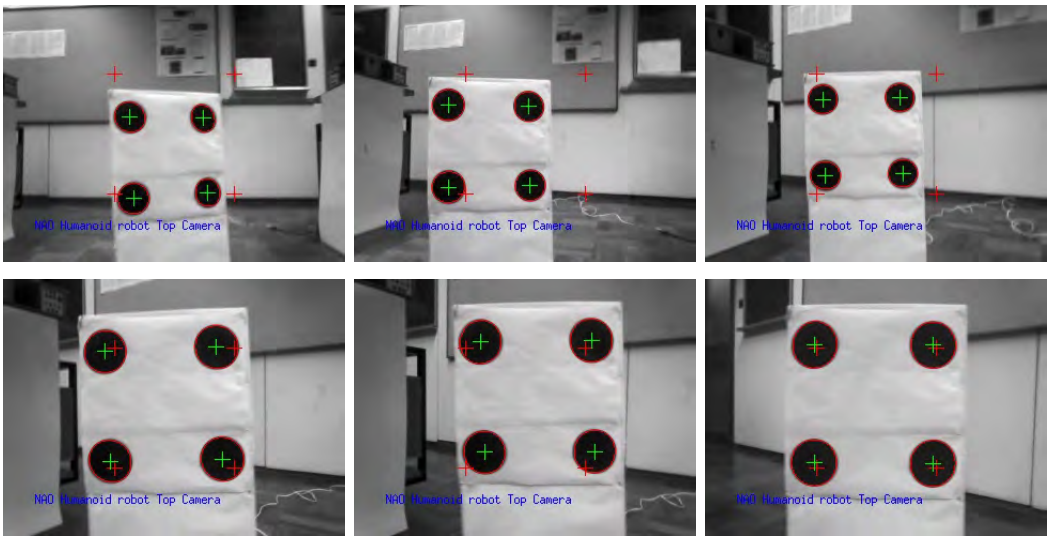


Figure 5.38: Sequence of image configurations during longitudinal and lateral tracking

The feed-forward control input used to compensated the target motion has also reduced the undesirable effect of the veering as can be seen in Figure 5.39 showing the image features trajectories from their initial to their final positions. This could be explained by the fact that the estimated compensation term comprises not only the target velocity, but also other uncertainties and disturbances affecting the system (see Appendix C.2 for more details).

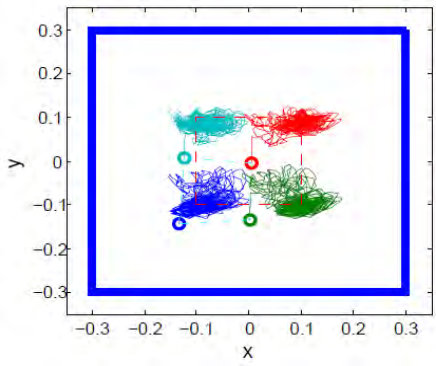
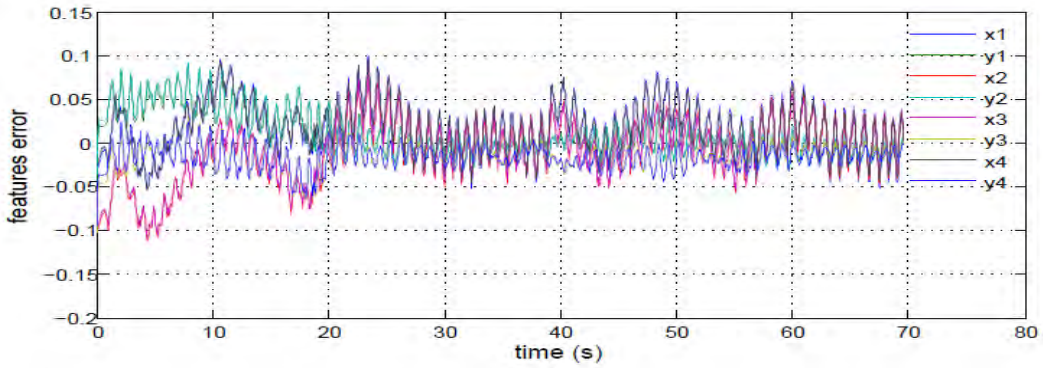
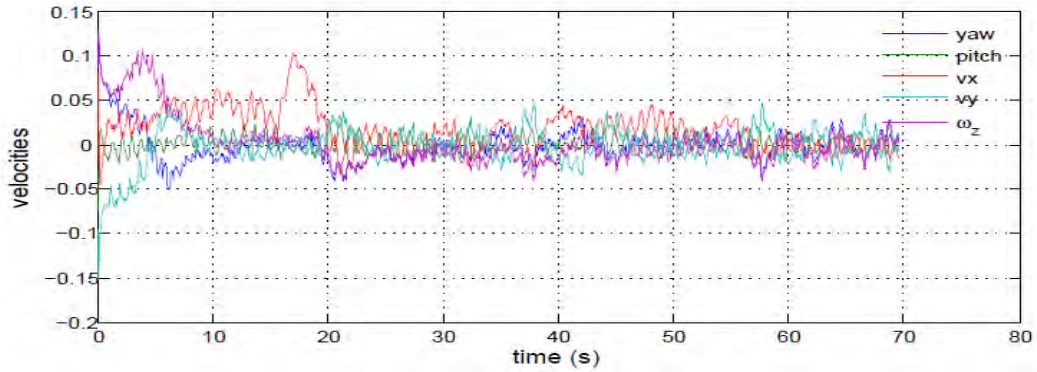


Figure 5.39: Image features trajectories during longitudinal and lateral tracking

The features error and the reference velocities are grouped in Figure 5.40. On velocities plots (Figure 5.41b), it can be seen that for the first 20 seconds, the robot walked towards the target and reoriented itself to face it and then started tracking with resultant velocity which now corresponds to v_x.



(a) Features error

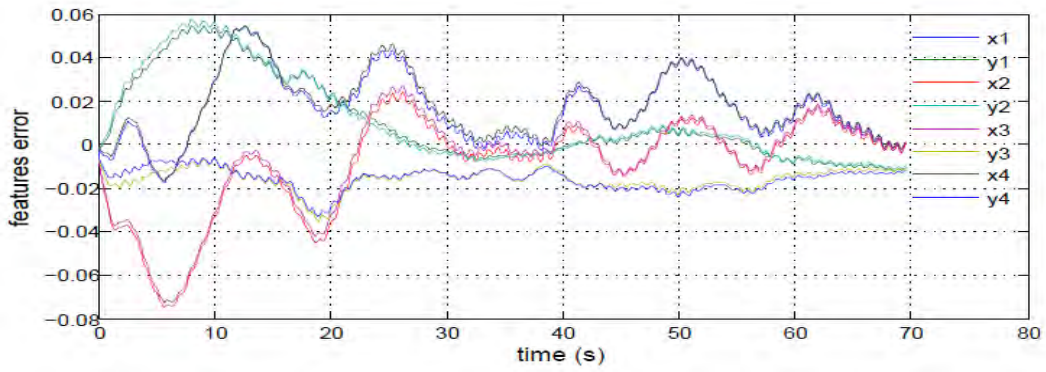


(b) Reference velocities

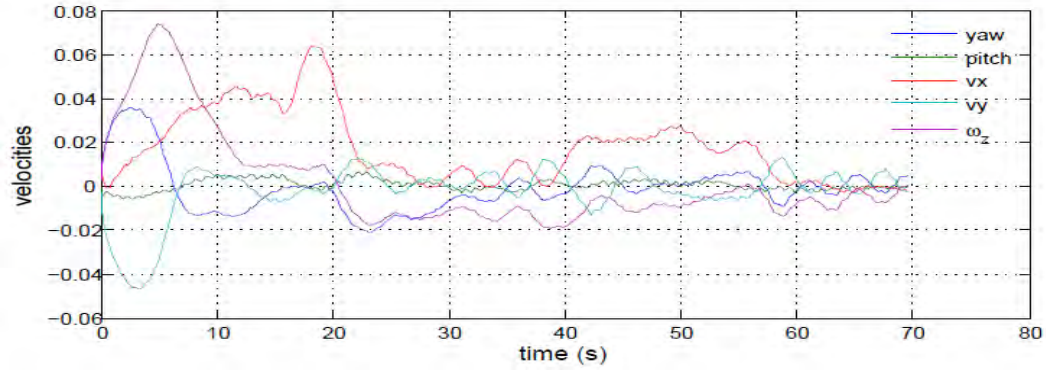
Figure 5.40: Experimental longitudinal and lateral tracking task

The features error decreases towards zero, of course with some oscillations especially for the horizontal components which are strongly affected by the sway motion. Additionally, the difficulty to maintain constant low speeds with the remote controlled car used in the experiments has contributed to the increase of these oscillations.

Filtering the results as shown in Figure 5.41 gives a better understanding of the system behavior during this task. Clearly, the tracking error has not been totally compensated as in simulations. This was in some how expected since the image and velocity measurements used to estimate the target motion were very noisy and discontinuous particularly those of the legs because of the switching. Nevertheless the robot managed to track the target and when the latter slowed down and stopped, the robot reached the desired configuration.



(a) Features Error



(b) Reference velocities

Figure 5.41: Filtered signals of the experimental longitudinal and lateral tracking task

5.4.4 Grasping

The grasping experiment was symbolically realized just to illustrate the usage of the humanoid's extra degrees of freedom to perform additional tasks while walking under reactive control scheme. Thus, under the same assumptions made in simulation regarding the object to be grasped (Section 5.3.4), instead of computing directly its pose from its appearance in image, the pose of the object was deducted from that of the square target and then used in a PBVS scheme.

The hand being totally out of the camera's field of view, joints positions and the kinematic model was used to compute the hand pose. Thus, this resulted in a hybrid visual-kinematic servoing. The position of the target to grasp with respect to the four points target is shown in Figure 5.42.

The results of this experiments are grouped in Figure 5.43, showing the features errors related to the positioning task (Figure 5.43a), its corresponding norm triggering the grasping task when it reaches a value of 0.02 (Figure 5.43b) and the error on the grasping pose (Figure 5.43c).

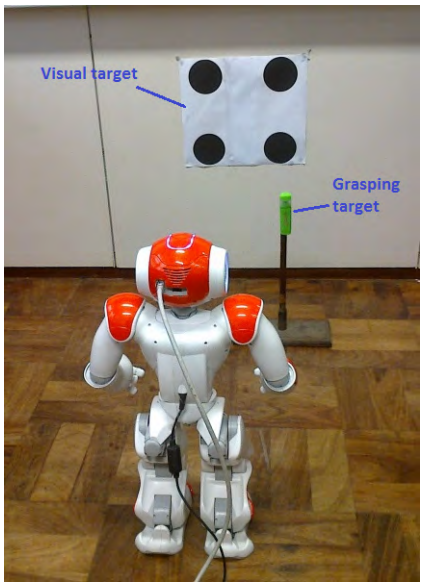
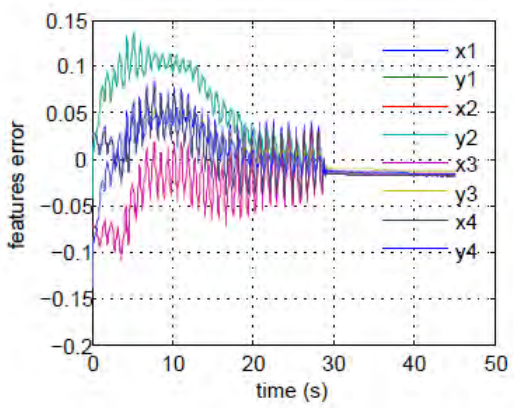
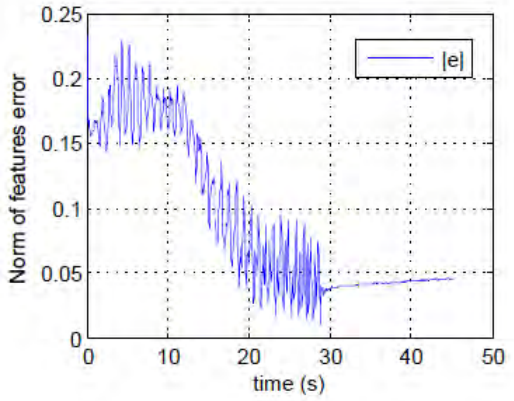


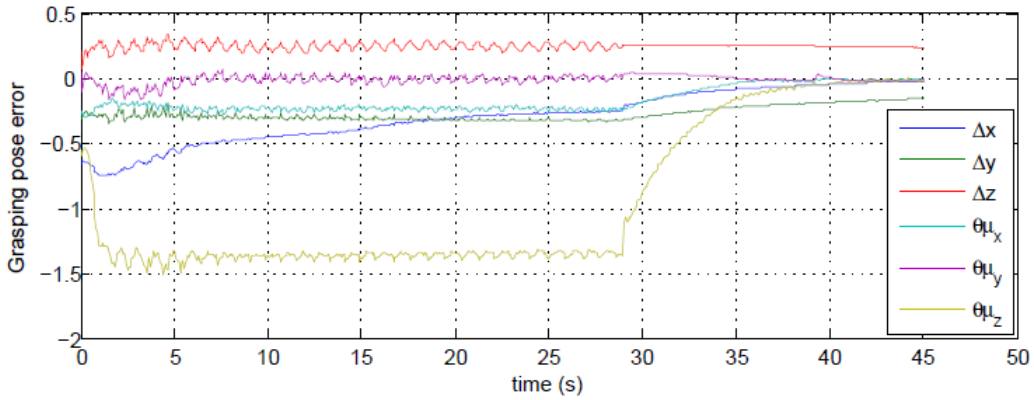
Figure 5.42: Target to grasp with respect to Four point target



(a) Features error



(b) Norm of features error



(c) Grasping pose error

Figure 5.43: Experimental visual servoing based object grasping: task errors

It can be noticed errors on two position variables. Since the position was projected onto the null-space of the orientation, it could only use the DoFs non-constrained by the orientation. Also, the calibration errors and the use of kinematic measurements in the PBVS loop contributed to this error. Despite those errors the norm of the grasping error was less than 0.05 the threshold for closing the hand as illustrated in Figure 5.44 showing a sequence of six images during this walk-to-grasp task.

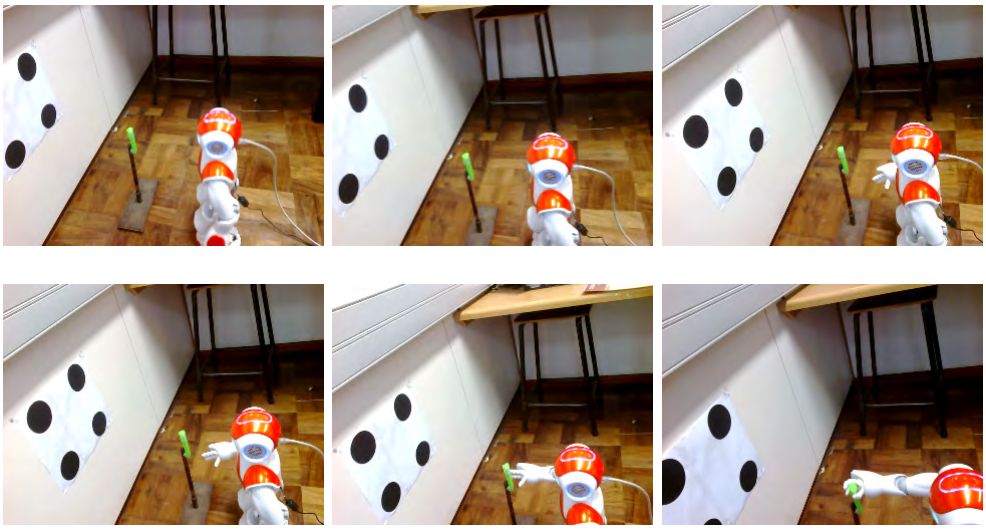
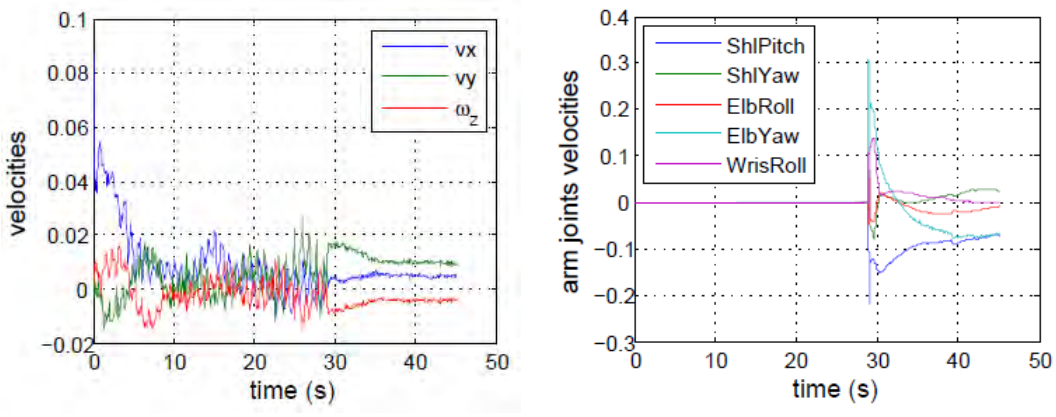


Figure 5.44: Sequence of image configurations during longitudinal and lateral tracking

The humanoid's base and arm's joints velocities are shown in Figure 5.45. The grasping was only triggered when the robot reached and stopped about the desired configuration as indicated by the constant features error after 30 s.



(a) Humanoid base's velocities

(b) Arm joints velocities

Figure 5.45: Humanoid's base and arm velocities during grasping experiment

It was noticed that triggering the grasping task during the walking was disturbing seriously, through the arm's motion, the robot's LIMP based pattern generator not only by changing the height of the CoM supposed to be constrained on a plane, but also by generating additional angular momentum about the CoM. This resulted in drifts of the robot from its desired trajectories. Such a disturbance can be confirmed by analyzing conjointly Figures 5.45a and 5.45b. Though stationary from the $30th$ second, the command velocities to the base are varying when the arm's velocity are high ($30\,s - 35\,s$). This means that the displacement of the base induced a displacement of the camera yielding thus a re-computation of the commands by the visual controller.

5.5 Conclusion

In this chapter, we have validated the proposed approach of visual servoing on a humanoid robot. We have shown that positioning tasks, target tracking tasks and additionally object grasping could be realized reactively by mapping the humanoid perception directly to its actions. Indeed, without any planning phase, the task to be performed by the humanoid robot was directly specified in the image of its embedded camera. Accounting for the humanoid structure, the control algorithm generated reference velocities for the robot, which once followed, led the humanoid to the desired configuration.

Going from simple towards more complex displacements, the reactive behavior of the humanoid robot was analyzed. It was seen that the positioning task has no particular requirement. The tracking task however, requires that the target's motion be compensated. For maneuvering target, for instance, this compensation is better realized with feed-forward control, which needs an estimate of the target motion. Then these analysis were confirmed experimentally on an actual robot.

Part II

Contribution to Dynamic Visual Servoing

Chapter 6

Dynamic Compensation in Visual Servoing

Introduction

In the first part of this thesis, visual servoing was formulated with a particular focus on convergence properties of robotic task in response to the perceived scene. The robot was implicitly assumed to be a perfect kinematic device, neglecting thus its dynamics. Such an approach is known as kinematic visual servoing [26]; it could be justified as long as the motions are slow and the accelerations are kept small [30, 56, 32, 33, 3, 15]. However, for fast motions or when high performance is required, the dynamics of the camera (essentially characterized by latency due to image acquisition, image processing and data transfer, and a nonlinear mapping from Cartesian to sensor space [31]) and the robot's dynamics (generally characterized by non-linearity and coupling in its model [124, 126, 169]) can no longer be neglected and have to be accounted for. That is the concern of dynamic visual servoing, which aims at modeling and compensating dynamic effects due to the vision and/or robot subsystem in order to improve the overall system's behavior [26].

In this chapter, we focus on dynamic compensation in visual servoing. The goal is to achieve fast positioning and tracking tasks, of course within the robot capabilities. Our methodology is to formulate the problem so as to improve first the reference commands generated from observed images and then to improve the robot's tracking performances for the generated references.

6.1 Previous Works

Several studies have addressed the problem of the dynamic visual servoing, having considered both modeling as well as compensation. Different approaches have been proposed and can be classified in linear and nonlinear-based approaches.

6.1.1 Linear Dynamic Visual Servoing

If linear or linearized models of the vision system and the robot are used, linear control theory techniques can thus be used. For instance, *Weiss et al.* [37], proposed an adaptive controller; the decoupling was ensured by assigning a single feature to each joint. Considering a tracking application, in [174], a self-turning controller is used to compensate the system dynamics, while a discrete autoregressive model is used to predicted the target's motion. In [175], the latter is estimated from optical flow computed by the sum of square difference (SSD). Controllers such as PI, poles placement, Linear Quadratic Gaussian (LQG), or adaptive controllers [176] were analyzed. Such controllers and a feed-forward one were also tested in [31, pp. 171 - 295], where stability issues and performance improvement in presence of sensor delay, multirate sampling and moving target were considered. In [177, 178, 179], controllability issues of a linearized system about its desired configuration was addressed and Linear Quadratic Regulator was designed. Generalized Predictive Control (GPC) was also employed, in [40, 41] for a 6 Dofs manipulator or in medical applications for filtering [180] and compensation [181].

Although linear methods have produced different results in dynamic visual servoing, their performances are restricted in regions around the considered operating points. Even the adaptive control, as proposed in [2], has shown its limits in dealing with a nonlinear and coupled model. To extend the performances to the whole robot's configuration space, piecewise linear modeling with gain scheduling was suggested [182].

6.1.2 Nonlinear Dynamic Visual Servoing

Another approach, to deal with dynamic visual servoing, is to exploit nonlinear control methods developed for robots. Thus, the problem can be formulated as follows:

Consider a robotic system whose dynamics can be written in the following form [126, p. 254]

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) = \Gamma \tag{6.1}$$

where $\mathbf{q} \in \mathbb{R}^n$ represents the joint vector, $\mathbf{D}(\mathbf{q}) \in \mathbb{R}^{n \times n}$ is the dynamic matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n \times n}$ is the matrix of centrifugal and Coriolis effects, $G(\mathbf{q})$ denotes the vector of gravity forces and $\Gamma \in \mathbb{R}^n$ is the vector of applied external forces/torques.

As previously, let $\boldsymbol{e}(\mathbf{q}, t) \triangleq \boldsymbol{\xi}(t) - \boldsymbol{\xi}^d$ be the visual task function representing the robotic task to be performed, where $\boldsymbol{\xi}(t)$ and $\boldsymbol{\xi}^d$ denote the current and desired image features vector, respectively. The first and second time-derivative of $\boldsymbol{e}(\mathbf{q}, t)$ can be shown to be given by [36]

$$\dot{\boldsymbol{e}}(\mathbf{q}, t) = \frac{\partial \boldsymbol{e}(\mathbf{q}, t)}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial \boldsymbol{e}(\mathbf{q}, t)}{\partial t} \tag{6.2}$$

$$\ddot{\boldsymbol{e}}(\mathbf{q},t) = \frac{\partial \boldsymbol{e}(\mathbf{q},t)}{\partial \mathbf{q}}\ddot{\mathbf{q}} + h(\mathbf{q},\dot{\mathbf{q}},t) \tag{6.3}$$

with $h(\mathbf{q},\dot{\mathbf{q}},t)$ being given by

$$h(\mathbf{q},\dot{\mathbf{q}},t) = \left[\dot{\mathbf{q}}^T \frac{\partial E_i^T(\mathbf{q},t)}{\partial \mathbf{q}}\dot{\mathbf{q}}\right] + 2\frac{\partial^2 \boldsymbol{e}(\mathbf{q},t)}{\partial \mathbf{q}\partial t} + \frac{\partial^2 \boldsymbol{e}(\mathbf{q},t)}{\partial t^2} \tag{6.4}$$

where E_i denotes the i^{th} row of the task Jacobian $\frac{\partial \boldsymbol{e}}{\partial \mathbf{q}}$.

To achieve the considered task, the robot has to be controlled such that $\boldsymbol{e}(\mathbf{q},t)$ is zeroed. One could choose an exponential decrease

$$\dot{\boldsymbol{e}}(\mathbf{q},t) + \Lambda_p \boldsymbol{e}(\mathbf{q},t) = 0 \tag{6.5}$$

where Λ_p is a proportional gain [30, 56, 32, 33, 3, 15]. One could also choose a second order decrease

$$\ddot{\boldsymbol{e}}(\mathbf{q},t) + \Lambda_v \dot{\boldsymbol{e}}(\mathbf{q},t) + \Lambda_p \boldsymbol{e}(\mathbf{q},t) = 0 \tag{6.6}$$

which accounts for the features acceleration and could be conveniently related to the robot dynamics. In such a case, the decrease of $\boldsymbol{e}(\mathbf{q},t)$ can thus be made fast and damped by tuning the gains Λ_v and Λ_p.

Depending on the space in which the control is realized, the above control problem can be formulated differently as shown in the sequel.

6.1.2.1 Image Space Control Formulation

Consider the case where the robot is controlled in the image space. Using (6.1) and (6.3), it can be shown [36] that the dynamics (6.1) can be expressed as function of $\boldsymbol{e}(\mathbf{q},t)$ as follows

$$\mathbf{D}(\mathbf{q})\left(\frac{\partial \boldsymbol{e}}{\partial \mathbf{q}}\right)^{-1}\ddot{\boldsymbol{e}} + N(\mathbf{q},\dot{\mathbf{q}},t) - \mathbf{D}(\mathbf{q})\left(\frac{\partial \boldsymbol{e}}{\partial \mathbf{q}}\right)^{-1} h(\mathbf{q},\dot{\mathbf{q}},t) = \Gamma \tag{6.7}$$

where $N(\mathbf{q},\dot{\mathbf{q}},t) \in \mathbb{R}^n$ is a vector lumping $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}$, $G(\mathbf{q})$ and any probable vector of friction forces.

Different possibilities exist to control the system (6.7). For instance, a feedback linearization can be used, where the control torque can be computed as

$$\Gamma_c = \mathbf{D}(\mathbf{q})\left(\frac{\partial \boldsymbol{e}}{\partial \mathbf{q}}\right)^{-1}\boldsymbol{u} + N - \mathbf{D}(\mathbf{q})\left(\frac{\partial \boldsymbol{e}}{\partial \mathbf{q}}\right)^{-1} h \tag{6.8}$$

where the control action $\boldsymbol{u}$ could be designed as a proportional-derivative law [36, 183]

$$\boldsymbol{u} = -K_D \left(K_P \boldsymbol{e} + \dot{\boldsymbol{e}} \right) \tag{6.9}$$

with G and D being positive matrices, in order to yields a closed loop of the form (6.6). Hence, substituting (6.8) with (6.9) in (6.7) gives the closed loop equation as

$$\ddot{\boldsymbol{e}} + K_D \left(\dot{\boldsymbol{e}} + K_P \boldsymbol{e} \right) = 0 \tag{6.10}$$

From (6.10), it is clear that if the acceleration is small and the gain K_D is high, this equation simplifies in the kinematic Equation (6.5).

Using this formulation, obtaining the closed loop Equation (6.10) requires that the control torques be computed directly from image measurements. Such an approach can be found in [184, 185], where the control torque was directly computed using the transpose Jacobian instead of its inverse as in Equation (6.8).

In general, the robot is already controlled and the vision system is used to generate the joints reference velocities or accelerations that will yield the desired decrease of $\boldsymbol{e}(\mathbf{q}, t)$.

6.1.2.2 Joint Space Control Formulation

Consider the case where the robot is controlled in the joint space. A linearization in joint space could be obtained by computing the control torque as follows [126, 186, 187]

$$\Gamma_c = \mathbf{D}(\mathbf{q})\boldsymbol{u} + N(\mathbf{q}, \dot{\mathbf{q}}, t) \tag{6.11}$$

with the control $\boldsymbol{u}$ chosen as

$$\boldsymbol{u} = \ddot{\mathbf{q}}_d - G(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) - K(\mathbf{q}_d - \mathbf{q}) \tag{6.12}$$

When (6.11) and (6.12) are substituted in (6.1), the resulting closed loop will be

$$(\ddot{\mathbf{q}} - \ddot{\mathbf{q}}_d) + G(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + K(\mathbf{q}_d - \mathbf{q}) = 0 \tag{6.13}$$

The second order decrease (6.6) can now be achieved by an appropriate design of $\ddot{\mathbf{q}}_d$. To that end, consider (6.3) with a motionless object ($\frac{\partial \boldsymbol{e}(\mathbf{q},t)}{\partial t} = 0$); it can be rewritten as

$$\begin{aligned} \ddot{\boldsymbol{e}}(t) &= \frac{\partial \boldsymbol{e}}{\partial \mathbf{q}} \ddot{\mathbf{q}} + \left(\frac{\partial \dot{\boldsymbol{e}}}{\partial \mathbf{q}} \right) \dot{\mathbf{q}} \\ &= \mathbf{J}_e \ddot{\mathbf{q}} + \dot{\mathbf{J}}_e \dot{\mathbf{q}} \end{aligned} \tag{6.14}$$

where $\mathbf{J}_e$ denotes the visual task Jacobian. If $\mathbf{J}_e^{-1}$ exists, from (6.14) $\ddot{\mathbf{q}} = \mathbf{J}_e^{-1}(\ddot{\boldsymbol{e}}(t) - \dot{\mathbf{J}}_e \dot{\mathbf{q}})$; hence, a general and realistic solution for $\ddot{\mathbf{q}}_d$ using (6.6) could be [188]

$$\ddot{\mathbf{q}}_d = -\widehat{\mathbf{J}}_e^{\dagger}(\Lambda_v \dot{\boldsymbol{e}}(t) + \Lambda_p \boldsymbol{e}(t)) + \widehat{\mathbf{J}}_e^{\dagger}\dot{\mathbf{J}}_e\dot{\mathbf{q}} \tag{6.15}$$

In practice, the uncertainties are not limited to the vision system; they also concern the robot. Thus, using this formulation, in [189, 190, 191] a robust adaptive compensation is proposed to deal with uncertainties associated with the camera and the robot dynamics. In a 3D tracking on mobile manipulator, adaptive control dealt with the dynamics uncertainties in [192]. In [193], it is extended to uncertainties associated with the kinematics, and with features depths in [194]. Dealing with the same problem, Lyapunov direct method is used to design a robust controller in [195, 196]; considering additionally the time-delay, *Li et al.*, [197] employed a sliding mode variable structure controller.

6.1.2.3 Operational Space Control Formulation

Consider now a robot controlled in the operational space. Using the mapping $\dot{\mathbf{x}} = \mathbf{J}_{\mathbf{q}}\dot{\mathbf{q}}$ and its time derivative, it can be shown that the robot dynamics in operational space is given by [125]

$$\mathbb{D}(\mathbf{q})\ddot{\mathbf{x}} + \mathbb{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \mathbb{G}(\mathbf{q}) = \mathsf{T} \tag{6.16}$$

where $\mathbb{D}(\mathbf{q})$, $\mathbb{C}(\mathbf{q}, \dot{\mathbf{q}})$, $\mathbb{G}(\mathbf{q})$ and T are the elements of the dynamic model view from the end-effector; they are given by

$$\begin{cases} \mathbb{D}(\mathbf{q}) & = \mathbf{J}_{\mathbf{q}}^{-T}\mathbf{D}(\mathbf{q})\mathbf{J}_{\mathbf{q}}^{-1} \\ \mathbb{C}(\mathbf{q}, \dot{\mathbf{q}}) & = \mathbf{J}_{\mathbf{q}}^{-T}\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{J}_{\mathbf{q}}^{-1} - \mathbb{D}(\mathbf{q})\mathbf{J}_{\mathbf{q}}\mathbf{J}_{\mathbf{q}}^{-1} \\ \mathbb{G}(\mathbf{q}) & = \mathbf{J}_{\mathbf{q}}^{-T}G(\mathbf{q}) \\ \mathsf{T} & = \mathbf{J}_{\mathbf{q}}^{T}\Gamma \end{cases} \tag{6.17}$$

Similarly to joint space, a feedback linearization could be realized by computing the control torque as follows [187, 124, pp. 195 - 199]

$$\mathsf{T} = \mathbb{D}(\mathbf{q})\mathbf{u} + \mathbb{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \mathbb{G}(\mathbf{q}) \tag{6.18}$$

with the control $\mathbf{u}$ design as

$$\mathbf{u} = \ddot{\mathbf{x}}_d - G(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) - K(\mathbf{x}_d - \mathbf{x}) \tag{6.19}$$

To achieve a second order decrease of the form (6.6), the task function can be expressed as function of the camera motion. Hence, its second order time-derivative will be given by

$$\ddot{\boldsymbol{e}}(t) = \mathbf{L}_e{}^c\dot{V} + \dot{\mathbf{L}}_e{}^cV \tag{6.20}$$

Thus, the desired camera acceleration yielding a decrease of the form (6.6) can be computed as [198]

$$\ddot{\mathbf{x}}_d = {}^c\dot{V}_d = -\widehat{\mathbf{L}}_e^{\dagger}(\Lambda_v \dot{\boldsymbol{e}}(t) + \Lambda_p \boldsymbol{e}(t)) + \widehat{\mathbf{L}}_e^{\dagger}\dot{\mathbf{L}}_e {}^cV \tag{6.21}$$

However, in order to avoid the computation of $\dot{\boldsymbol{e}}(t)$, using sequential acquisition of region of interest in image, the following law was proposed in [42]

$$\ddot{\mathbf{x}}_d = {}^c\dot{V}_d = -[\ K_p \quad K_v\ ]\widehat{\mathbf{L}}_{sq}^{\dagger}\boldsymbol{e}(t) + \widehat{\mathbf{L}}_{sq\_l}^{\dagger}\dot{\boldsymbol{\xi}}^d \tag{6.22}$$

where $\mathbf{L}_{sq} \in \mathbb{R}^{2n\times 12}$ is a interaction matrix relating the camera velocity and acceleration to the time-derivative of a set of n image points and $\mathbf{L}_{sq\_l}^{\dagger} \in \mathbb{R}^{6\times 2n}$ is the lower sub-matrix of $\mathbf{L}_{sq}^{\dagger}$. In this method, the required features variation is implicitly contained in the sequence of visual features (more details can be found in [42]).

Remark. The nonlinear approaches described above are more effective than their linear counterpart [184, 189, 198]. However, in their implementation, all these methods, except law (6.22), require the time-derivative $\dot{\boldsymbol{e}}(t)$, which is not directly measurable. It has therefore to be estimated or derived from the features $\boldsymbol{e}(t)$. For instance, it was approximated by $\dot{\widehat{\boldsymbol{e}}}(t) = \widehat{\mathbf{J}}_e\dot{\mathbf{q}} + \widehat{\frac{\partial \boldsymbol{e}(t)}{\partial t}}$ in [36] and [198]; in [183] it was estimated with an adaptive observer of the form $\dot{\widehat{\boldsymbol{e}}}(t) = \mathbf{J}_e\dot{\mathbf{q}} + \mathbf{L}_e W\widehat{\boldsymbol{\theta}} + H(\widehat{\boldsymbol{e}} - \boldsymbol{e})$ with the updating law $\dot{\widehat{\boldsymbol{\theta}}} = -W^T\mathbf{L}_e^T P(\widehat{\boldsymbol{e}} - \boldsymbol{e})$, where $W\widehat{\boldsymbol{\theta}}$ represents a linear parametrization of the object motion, and H and P are gain matrices. In these methods however, one needs to know well the Jacobian of the task in addition to velocity measurement. Furthermore, if the camera velocity could be fairly estimated, it is not the same for the object velocity, which has to be extracted from noisy and delayed images.

In approaches attempting to achieve an exponential decrease of $\boldsymbol{e}(t)$ [189, 190, 193, 194, 196, 197, 192], the features are indirectly differentiated through the velocity ($\dot{\mathbf{q}}_d = -\Lambda_p\widehat{\mathbf{J}}_e^{\dagger}\boldsymbol{e}(t)$) while computing the required reference acceleration given by $\ddot{\mathbf{q}}_d = -\Lambda_p(\widehat{\mathbf{J}}_e^{\dagger}\dot{\boldsymbol{e}}(t) - \dot{\widehat{\mathbf{J}}}_e^{\dagger}\boldsymbol{e}(t))$. This differentiation of features has the downside of amplifying noise.

Thus, improving the overall system performances should not be limited solely to the robot, the generation of reference trajectories, usually based on an exponential decay of the task function, should also be improved.

Indeed, it was noted in [42] that the first order exponential decay, is not the fastest response. The authors also argue that the resulting velocity based control is not very suitable for better dynamic compensation; acceleration based control was deemed adequate [199]. Such control was also suggested in [198] and [188].

6.2 Proposed Approach and Contribution

In this chapter we propose on one hand, a second order decrease of the task function with a PD type control law. The reference acceleration necessary for dynamic compensation is easily determined without neither computing the optical flow nor even estimating it as in [198] and [188].
The key idea is to use, in classical visual servoing law, a first order low-pass filter on the task function prior the computation of the reference velocity. This yields faster and damped closed loop response in the image for both acceleration and velocity based control.

Inspired by the works [194], [196] and [192], we propose, on the other hand, an adaptive and robust compensation scheme associated with the proposed visual control law. The stability and convergence analysis is performed by invoking the Lyapunov direct method.

The developed dynamic compensation uses reference velocity and / or acceleration, which presuppose the existence of compatible inputs at the robot level. To render it applicable to cases where the only allowed input is the reference position, we present next a transformation to convert the resulting torque control into reference position.

6.3 Robot Dynamic Remodeling

Robots are generally manufactured and sold with built-in low-level controllers, often PID, dealing with low level inputs such as actuators voltage. The user is then limited to specify reference position or velocity inputs and often does not control directly the low-level inputs. Thus, the actual control variables for the user are the reference signals. Therefore, modeling the robot dynamics viewed from those reference signals becomes more appropriate [200]. This is particularly true when in a hierarchical control architecture, high level commands are sent at lower level to be executed; the robot dynamics viewed from these commands is different from that viewed from the voltage inputs. Hence, the complete model should account for the dynamics of the low-level controller.

To that end, let us consider the dynamic model of a robot manipulator (6.1) and assume that the robot actuators are armature current DC motors, whose dynamics is given by [126, p. 207]

$$J_{m_k}\ddot{\theta}_{m_k} + (B_{m_k} + \frac{K_{b_k}K_{m_k}}{R_k})\dot{\theta}_{m_k} = \frac{K_{m_k}}{R_k}V_k - \frac{\tau_{l_k}}{r_k} \tag{6.23}$$

where θ_{m_k} is the position angle of the motor shaft, J_{m_k} and B_{m_k} represent respectively, the inertia and damping of both the motor and the gear. K_{b_k} and K_{m_k} are the back emf and torque constants, respectively. V_k and R_k denote the armature voltage and resistance, respectively. τ_{l_k} is the load torque, while r_k is the gear ratio. The actuator dynamics can be rewritten in joint

space given that $\theta_{m_k} = r_k q_k$. Thus, Equation (6.23) becomes

$$r_k^2 J_{m_k} \ddot{q}_k + r_k^2 B_k \dot{q}_k = r_k \frac{K_{m_k}}{R_k} V_k - \tau_k \tag{6.24}$$

with $B_k \triangleq B_{m_k} + \frac{K_{b_k} K_{m_k}}{R_k}$. Combining Equation (6.24) and Equation (6.1) while written in the developed form, gives the overall dynamics of the system as [126, p. 206]

$$r_k^2 J_{m_k} \ddot{q}_k + \sum_{j=1}^{n} d_{kj} \ddot{q}_j + r_k^2 B_k \dot{q}_k + \sum_{i,j=1}^{n} c_{ijk} \dot{q}_i \dot{q}_j + \phi_k = r_k \frac{K_{m_k}}{R_k} V_k \tag{6.25}$$

To be realistic, a friction term and a bounded disturbance term are incorporated to the ideal model (6.25). Hence, considering all joints, the overall system's dynamics can be written in the following compact form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) + F(\dot{\mathbf{q}}) + T_d = \boldsymbol{u} \tag{6.26}$$

where

$$\begin{aligned} \mathbf{M}(\mathbf{q}) &= \mathbf{D}(\mathbf{q}) + diag(r_k^2 J_{m_k}) \\ &= \mathbf{D}(\mathbf{q}) + \mathbf{J} \\ \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) &= \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{B} \\ \boldsymbol{u} &= diag\left(r_k \frac{K_{m_k}}{R_k}\right) V \end{aligned} \tag{6.27}$$

with $F(\dot{\mathbf{q}})$ and T_d denoting the frictional and bounded disturbance torque, respectively.

Next, we consider the case of a lower-level PD controller. Similarly, this approach can also be extended to the PID case.

6.3.1 Low Level PD Controller

The control variable at the actuator level being the voltage, consider the PD control law of the following form

$$V_k = k_{p_k} \left(q_{lsp_k} - q_k\right) - k_{d_k} \dot{q} \tag{6.28}$$

where the subscript $_{lsp}$ stands for low-level set-point, while k_{p_k} and k_{d_k} denote the proportional and derivative gains. Using Equation (6.27) and (6.28), the input torque u_k to the combined dynamics is then given by

$$u_k = r_k \frac{k_{p_k} K_{m_k}}{R_k} q_{lsp_k} - r_k \frac{k_{p_k} K_{m_k}}{R_k} q_k - r_k \frac{K_{m_k} k_{d_k}}{R_k} \dot{q}_k \tag{6.29}$$

From Equation (6.26) and (6.29) it can be shown that the resulting closed loop equation of the robot dynamics will be given by

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) + F(\dot{\mathbf{q}}) + T_d = \mathbf{P}\mathbf{q}_{lsp} - \mathbf{P}\mathbf{q} - \mathbf{Q}\dot{\mathbf{q}} \tag{6.30}$$

where the right-hand side of Equation (6.30), $\mathbf{P}\mathbf{q}_{lsp} - \mathbf{P}\mathbf{q} - \mathbf{Q}\dot{\mathbf{q}} = \boldsymbol{u}$, with $\mathbf{P} = diag(r_k \frac{k_{p_k} K_{m_k}}{R_k})$ and $\mathbf{Q} = diag(r_k \frac{K_{m_k} k_{d_k}}{R_k})$. In order to reflect the robot dynamics viewed from the reference signal, as proposed in [200, 192], Equation (6.30) can be rewritten as follows

$$\overline{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \overline{\mathbf{N}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \overline{G}(\mathbf{q}) + \overline{\Phi}(\mathbf{q}) = \mathbf{q}_{lsp} \tag{6.31}$$

where

$$\begin{cases} \overline{\mathbf{M}}(\mathbf{q}) & = \mathbf{P}^{-1}\mathbf{M}(\mathbf{q}) \\ \overline{\mathbf{N}}(\mathbf{q},\dot{\mathbf{q}}) & = \mathbf{P}^{-1}\left(\mathbf{N}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{Q}\right) \\ \overline{G}(\mathbf{q}) & = \mathbf{P}^{-1}G(\mathbf{q}) \\ \overline{\Phi}(\mathbf{q},\dot{\mathbf{q}}) & = \mathbf{P}^{-1}(F(\dot{\mathbf{q}}) + T_d + \mathbf{P}\mathbf{q}) \end{cases} \tag{6.32}$$

Note that (6.31) can also be written in terms of the error $\Delta\mathbf{q}_{lsp} = (\mathbf{q}_{lsp} - \mathbf{q})$, which can be seen as a *relative reference* needed to bring the robot from its current state to the desired one. Hence, we have

$$\overline{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \overline{\mathbf{N}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \overline{G}(\mathbf{q}) + \breve{\Phi}(\mathbf{q}) = \Delta\mathbf{q}_{lsp} \tag{6.33}$$

where $\breve{\Phi}(\mathbf{q}) \triangleq \mathbf{P}^{-1}(F(\dot{\mathbf{q}}) + T_d)$. The relative reference formulation offers some advantages, especially in the implementation.

Note. Given the models (6.31) and (6.33), which represent the controlled robot's dynamic models as seen by a high-level controller (e.g. a visual servo controller or dynamic trajectory planner), a computed-torque like control scheme [126, p. 261] [201, p. 402] can now be applied to compute the robot's low-level reference input $\mathbf{q}^*_{lsp}$ or $\Delta\mathbf{q}^*_{lsp}$ that will drive the robot's states to follow a desired trajectory[192].

$$\mathbf{q}^*_{lsp} = \widehat{\overline{\mathbf{M}}}\hat{\ddot{\mathbf{q}}}_{ref} + \widehat{\overline{\mathbf{N}}}\dot{\mathbf{q}}_{ref} + \widehat{\overline{G}} + \widehat{\overline{\Phi}} \tag{6.34}$$

where "hat" symbol means approximate values and $\dot{\mathbf{q}}_{ref}$ and $\ddot{\mathbf{q}}_{ref}$ denote the reference velocity and acceleration, respectively. They might result from a high-level controller in a hierarchical scheme [193, 196, 192].

Remark. The performances of the resulting closed loop will depend on the design of $\mathbf{q}^*_{lsp}$, which represents here a generic robot motion controller. Appropriate choice of $\mathbf{q}^*_{lsp}$'s variables produces well known robot motion controllers [187], whose stability and convergence properties

are firmly established. For instance:

- if $\widehat{\overline{\mathbf{M}}}(\mathbf{q})$, $\widehat{\mathbf{N}}(\mathbf{q},\dot{\mathbf{q}})$, $\widehat{G}(\mathbf{q})$ and $\widehat{\Phi}$ are selected as their exact values and $\hat{\dot{\mathbf{q}}}_{ref} = \dot{\mathbf{q}}$, $\hat{\ddot{\mathbf{q}}}_{ref} = \ddot{\mathbf{q}}_d + K_v\dot{e}_q + K_p e_q$, with $e_q \triangleq \mathbf{q}_d - \mathbf{q}$, one obtains a similar controller to the so called "Computed-torque control" [124, p. 191], [186, p. 227].

- if $\hat{\dot{\mathbf{q}}}_{ref} = \dot{\mathbf{q}}$, $\hat{\ddot{\mathbf{q}}}_{ref} = \ddot{\mathbf{q}}_d + K_v\dot{e}_q + K_p e_q + \Delta\hat{\ddot{\mathbf{q}}}_{ref}$, where $\Delta\hat{\ddot{\mathbf{q}}}_{ref}$ is an additional control input designed to overcome uncertainties in the system such that

$$\Delta\hat{\ddot{q}}_{ref} = \begin{cases} -\rho(e,t)\frac{B^T P e}{\|B^T P e\|} & ;\, if \;\; \left\|B^T P e\right\| \neq 0 \\ 0 & ;\, if \;\; \left\|B^T P e\right\| = 0 \end{cases}$$

  where $e^T \triangleq [e_q^T,\, \dot{e}_q^T]^T$, $B = (0,\, \mathbf{I}_n)^T$, $P \in \mathbb{R}^{2n\times 2n}$ is a symmetric positive-definite matrix, the resulting robust controller is known as "Computer-torque-like control with variable-structure compensation" (see for more details [187][126, p. 263] and the references therein).

- if the variables are chosen such that $\widehat{\dot{\Phi}} = \widehat{\dot{\Phi}} + K_D\boldsymbol{s}$, $\hat{\dot{\mathbf{q}}}_{ref} = \dot{\mathbf{q}}_d + \Lambda e_q$, $\hat{\ddot{\mathbf{q}}}_{ref} = \ddot{\mathbf{q}}_d + \Lambda\dot{e}_q$, where $\boldsymbol{s} = \dot{e}_q + \Lambda e_q$ with Λ a positive-definite matrix, and if in addition it is used an adaptive law of the form $\dot{\hat{a}} = -L_a Y^T(\mathbf{q},\dot{\mathbf{q}},\hat{\dot{\mathbf{q}}}_{ref},\hat{\ddot{\mathbf{q}}}_{ref})\boldsymbol{s}$ where $Y(\mathbf{q},\dot{\mathbf{q}},\hat{\dot{\mathbf{q}}}_{ref},\hat{\ddot{\mathbf{q}}}_{ref}) \in \mathbb{R}^{n\times m}$ is the regressor of the robot dynamics, $\hat{a} \in \mathbb{R}^m$ is the vector of system's estimated parameters, L_a a positive-definite matrix, then the resulting controller is known as "Adaptive inertia-related control" [201, pp. 403 - 406], [187].

Next, we present our approach yielding a second order decrease of $\boldsymbol{e}(t)$ without the need to compute $\dot{\boldsymbol{e}}(t)$ nor even to estimate it.

6.4 Proposed Dynamic Visual Servoing Laws

In this section, we propose a novel approach allowing to achieve a second order decrease of the task function with the benefit of being fast and damped as in (6.10), without any need to compute explicitly the required time derivative of the image features nor even to approximate it. We start by formulating the problem for a single feature and afterward, we generalize the formulation to multiple image features.

6.4.1 Single Image Feature

Consider the closed loop equation of a dynamic visual servoing system under a feedback linearization control scheme [36]. Let us assume that the task function is made of a single image feature, that is $e(t)$ is scalar. Hence, the closed loop Equation (6.10) can be written as

$$\ddot{e}_i(t) + \alpha_v\dot{e}_i(t) + \alpha_p e_i(t) = 0 \tag{6.35}$$

where α_p and α_v denote the proportional and derivative gains, respectively. This function can be rewritten as

$$\frac{d}{dt}\dot{e}_i(t) + \alpha_v \dot{e}_i(t) = -\alpha_p e_i(t) \tag{6.36}$$

Applying the Laplace Transform to (6.36), it can be shown that

$$\dot{e}_i(s) = -\frac{\alpha_p e_i(s)}{(s + \alpha_v)} \tag{6.37}$$

Since the time derivative of feature for a motionless object is given by

$$\dot{e}_i(t) = L_{e_i}\, {}^cV \tag{6.38}$$

we can infer that choosing the ideal control law

$${}^cV = -L_{e_i}^{\dagger} \frac{\alpha_p e_i(t)}{(s + \alpha_v)} \tag{6.39}$$

will results in the closed loop Equation (6.35) provided that $L_{e_i} \cdot L_{e_i}^{\dagger} = 1$.

Remark. Although simple, the result in Equation (6.39) is very interesting, since it shows that there is no need to compute the time-derivative of the image feature (optical flow) in order to implement a dynamic visual servoing. *Filtering* the image feature prior the computation of the velocity will result in a second order decrease of the task function. This technique has a twofold advantage: not only it allows fast and damped response, it also reduces noise as opposed to the usage of the time-derivative of image feature [189, 190, 193, 194, 196, 197, 192].

6.4.2 Multiple Image Features Case

Consider now the case where we have multiple image features and only an approximate, $\widehat{\mathbf{L}}_e$, of the interaction matrix. Following Equation (6.39), the velocity vector will now be given by

$${}^cV = -\widehat{\mathbf{L}}_e^{\dagger} \left[s\mathbf{I} + \Lambda_v\right] \Lambda_v(\Lambda_p \boldsymbol{e}(t)) \tag{6.40}$$

where Λ_p and Λ_v are constant gain matrices. The resulting closed loop equation can be obtained by substituting (6.40) in the vector form of Equation (6.38), which gives

$$\dot{\boldsymbol{e}}(t) = -\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger} \left[s\mathbf{I} + \Lambda_v\right]^{-1} \Lambda_v(\Lambda_p \boldsymbol{e}(t)) \tag{6.41}$$

To derive the explicit expression of the closed loop, consider a generalized inverse $(\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^+$ such that $(\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^+ (\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger}) = \mathbf{I}$. Then, multiplying both side of Equation (6.41) successively by $(\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^+$ and $[sI + \Lambda_v]$ gives

$$sI \left((\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^+ \dot{\boldsymbol{e}}(t) \right) + \Lambda_v \left((\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^+ \dot{\boldsymbol{e}}(t) \right) = -\Lambda_v \left(\Lambda_p \boldsymbol{e}(t) \right) \tag{6.42}$$

when substituting s for $\frac{d}{dt}$, this equation can be rewritten as

$$(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)^+\ddot{\boldsymbol{e}}(t) + \mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^\dagger)^+\dot{\boldsymbol{e}}(t) + \Lambda_v(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)^+\dot{\boldsymbol{e}}(t) = -\Lambda_v\Lambda_p\boldsymbol{e}(t) \tag{6.43}$$

Given that $(\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^\dagger)^+ = -(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)^+(\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^\dagger)(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)^+$, multiplying again both sides of (6.43) by $(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)$ yields

$$\ddot{\boldsymbol{e}}(t) - (\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^\dagger)(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)^+\dot{\boldsymbol{e}}(t) + \Lambda_v\dot{\boldsymbol{e}}(t) = -\Lambda_v\Lambda_p(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)\boldsymbol{e}(t) \tag{6.44}$$

Finally, grouping similar terms gives

$$\ddot{\boldsymbol{e}}(t) + \left(\Lambda_v - (\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^\dagger)(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)^+\right)\dot{\boldsymbol{e}}(t) + \Lambda_v\Lambda_p(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger)\boldsymbol{e}(t) = 0 \tag{6.45}$$

Equation (6.45) represents the closed loop equation of the visual servoing system under the control law (6.40). Clearly, it shows that if the approximate $\widehat{\mathbf{L}}_e^\dagger$ is not too coarse, such that $(\mathbf{L}_e\widehat{\mathbf{L}}_e^\dagger) \approx \mathbf{I}$ and thereby its time-derivative $(\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^\dagger) \approx 0$, Equation (6.45) will result in the following ideal closed loop equation

$$\ddot{\boldsymbol{e}}(t) + \Lambda_v\dot{\boldsymbol{e}}(t) + \Lambda_v\Lambda_p\boldsymbol{e}(t) = 0 \tag{6.46}$$

We call (6.46) ideal because it is linear and totally decoupled. In state-space, (6.46) can be written as

$$\frac{d}{dt}\boldsymbol{\epsilon} = A\boldsymbol{\epsilon} \tag{6.47}$$

where

$$\boldsymbol{\epsilon} = \begin{bmatrix} \boldsymbol{e} \\ \dot{\boldsymbol{e}} \end{bmatrix} \quad \text{and} \quad A = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\Lambda_v\Lambda_p & -\Lambda_v \end{bmatrix}$$

Since Λ_p and Λ_v are positive-definite matrices, which give a Hurwitz A matrix, Equation (6.46) is known to be global exponentially stable [187]. In other words, if a Lyapunov function of the form $\mathcal{L}(\boldsymbol{\epsilon}, t) = \boldsymbol{\epsilon}^T P\boldsymbol{\epsilon}$ is defined, with P and Q two positive-definite matrices satisfying the Lyapunov equation $A^TP + PA = -Q$ [201, pp. 81 - 93], then $\mathcal{L}(\boldsymbol{\epsilon}, t)$ will exponentially decreases such that

$$\mathcal{L}(\boldsymbol{\epsilon}, t) = \mathcal{L}(\boldsymbol{\epsilon}, 0) \cdot e^{-(\frac{\lambda_{min}(Q)}{\lambda_{max}(P)})t} \tag{6.48}$$

where $\lambda_{max}(P)$ and $\lambda_{min}(Q)$ are the maximum and minimum eigenvalues of P and Q, respectively.

6.4.3 Stability and Convergence Analysis

The stability analysis of the proposed dynamic visual servoing law (6.40) whose closed loop equation is (6.45) will be carried out in two ways. First, we will analyze a particular case and then we will try to generalize the analysis.

6.4.3.1 Particular Analysis

The first approach is inspired by the ideal case and assumes that $\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger} > 0$, which is a common assumption in stability analysis of most visual servoing laws [36, 30, 3, 15]. The closed loop (6.45) can be written is state space as

$$\frac{d}{dt}\boldsymbol{\epsilon} = A_S\boldsymbol{\epsilon} \tag{6.49}$$

where

$$A_S = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\Lambda_v\Lambda_p(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger}) & -(\Lambda_v - (\dot{\mathbf{L}_e}\widehat{\mathbf{L}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+}) \end{bmatrix} \tag{6.50}$$

If in addition, it is assumed that Λ_v is chosen such that $\Lambda_{v_i} \gg \sigma_{max}\left((\dot{\mathbf{L}_e}\widehat{\mathbf{L}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+}\right)$, where σ_{max} denotes the maximum singular value, then the matrix A_S will admit strictly negative eigenvalues and will therefore be Hurwitz. Thus, similarly to the ideal case (6.46), one can find matrices $P_S > 0$ and $Q_S > 0$ such that $A_S^T P_S + P_S A_S = -Q_S$. Considering the Lyapunov function

$$\mathcal{L}(\boldsymbol{\epsilon}, t) = \boldsymbol{\epsilon}^T P_S \boldsymbol{\epsilon} > 0, \tag{6.51}$$

its time-derivative is given by

$$\dot{\mathcal{L}}(\boldsymbol{\epsilon}, t) = \boldsymbol{\epsilon}^T P_S \dot{\boldsymbol{\epsilon}} + \dot{\boldsymbol{\epsilon}}^T P_S \boldsymbol{\epsilon} \tag{6.52}$$

Substituting (6.50) in (6.52) and grouping similar term gives

$$\dot{\mathcal{L}}(\boldsymbol{\epsilon}, t) = \boldsymbol{\epsilon}^T (P_S A_S + A_S^T P_S)\boldsymbol{\epsilon} = -\boldsymbol{\epsilon}^T Q_S \boldsymbol{\epsilon} < 0 \tag{6.53}$$

Under the above assumption we can conclude that the proposed law is stable and convergent. However, the positive-definiteness of $\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger}$ cannot always be guaranteed, particularly in IBVS [56, 67]; therefore, we can only conclude on local asymptotic stability.

6.4.3.2 General Analysis

In a more general sense, in order to determine the conditions on the gains that will guarantee the stability and the convergence of the system, the stability and convergence analysis of (6.45) can be carried out by using perturbation theory concepts [202, chap. 9]. Hence, let us rewrite Equation (6.49) as follows

$$\begin{aligned} \frac{d}{dt}\boldsymbol{\epsilon} &= f(\boldsymbol{\epsilon}, t) + g(\boldsymbol{\epsilon}, t) \\ &= A_N\boldsymbol{\epsilon} + \delta A_P \boldsymbol{\epsilon} \end{aligned} \tag{6.54}$$

where

$$
\begin{aligned}
A_N &= \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\Lambda_v\Lambda_p & -\Lambda_v \end{bmatrix} \\
\delta A_P &= \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \Lambda_v\Lambda_p(\mathbf{I} - \mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger}) & \Lambda_v(\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+} \end{bmatrix}
\end{aligned}
\tag{6.55}
$$

with $f(\boldsymbol{\epsilon}, t) = A_N\boldsymbol{\epsilon}$ representing the nominal model or the non-perturbed system, which is known to be stable (A_N is Hurwitz), and $g(\boldsymbol{\epsilon}, t) = \delta A_P\boldsymbol{\epsilon}$ represents the perturbation.

When the nominal system settles or reaches its equilibrium point ($\boldsymbol{\epsilon} = 0$), the perturbation vanishes ($g(\mathbf{0}, t) = 0$). Therefore, the stability of the overall system can be proven using the theorem of dynamic system under vanishing perturbation [202, pp. 339 - 342]. This theorem states that if for the nominal system, a candidate Lyapunov function $\mathcal{L}_N(\boldsymbol{\epsilon}, t)$ can be found such that

$$
c_1 \|\epsilon\|^2 \leq \mathcal{L}_N(\boldsymbol{\epsilon}, t) \leq c_2 \|\epsilon\|^2 \tag{6.56}
$$

$$
\dot{\mathcal{L}}_N(\boldsymbol{\epsilon}, t) = \frac{\partial \mathcal{L}_N}{\partial t} + \frac{\partial \mathcal{L}_N}{\partial \boldsymbol{\epsilon}} f(\boldsymbol{\epsilon}, t) \leq -c_3 \|\epsilon\|^2 \tag{6.57}
$$

$$
\left\| \frac{\partial \mathcal{L}_N}{\partial \boldsymbol{\epsilon}} \right\| \leq c_4 \|\epsilon\| \tag{6.58}
$$

and if

$$
\|g(\boldsymbol{\epsilon}, t)\| \leq \gamma \|\epsilon\|, \forall t \geq 0, \tag{6.59}
$$

where c_1, c_2, c_3, c_4 and γ are positive constants, with

$$
\gamma < \frac{c_3}{c_4} \tag{6.60}
$$

Then the system's origin ($\boldsymbol{\epsilon} = 0$) is globally exponentially stable [201, p. 50].

To analyze the closed loop stability under this framework, consider the candidate Lyapunov function

$$
\mathcal{L}_N(\boldsymbol{\epsilon}) = \boldsymbol{\epsilon}^T P_N \boldsymbol{\epsilon} > 0 \tag{6.61}
$$

where as previously, $P_N > 0$ and $Q_N > 0$ such that $P_N A_N + A_N^T P_N = -Q_N$. We can write

$$
\lambda_{min}(P_N) \|\epsilon\|^2 \leq \mathcal{L}_N(\boldsymbol{\epsilon}) \leq \lambda_{max}(P_N) \|\epsilon\|^2 \tag{6.62}
$$

The time-derivative of $\mathcal{L}_N(\boldsymbol{\epsilon})$ is given by

$$\dot{\mathcal{L}}_N(\boldsymbol{\epsilon}) = \boldsymbol{\epsilon}^T(P_N A_N + A_N^T P_N)\boldsymbol{\epsilon} = -\boldsymbol{\epsilon}^T Q_N \boldsymbol{\epsilon} \leq -\lambda_{min}(Q_N)\,\|\epsilon\|^2 \tag{6.63}$$

we also have

$$\left\| \frac{\partial \mathcal{L}_N}{\partial \boldsymbol{\epsilon}} \right\| = \boldsymbol{\epsilon}^T P_N + P_N \boldsymbol{\epsilon} \leq 2\lambda_{max}(P_N)\,\|\epsilon\| \tag{6.64}$$

To conclude that the system (6.54) is stable, the perturbation

$$g(\boldsymbol{\epsilon}, t) = \delta A_P \boldsymbol{\epsilon} = \left(\Lambda_v \Lambda_p (\mathbf{I} - \mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger)\right) \boldsymbol{\epsilon}_1 + \left(\Lambda_v (\dot{\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger})(\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger)^+\right) \boldsymbol{\epsilon}_2 \tag{6.65}$$

according to (6.59), has to be bounded as well. Hence, from (6.65), we can write the following inequality

$$\begin{aligned} \|g(\boldsymbol{\epsilon}, t)\| &\leq max\,\|\delta A_P\|\,\|\boldsymbol{\epsilon}\| \\ &\leq \lambda_{max}(\Lambda_v)\,\|\boldsymbol{\epsilon}\| \left(\left\| \Lambda_p (\mathbf{I} - \mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger) \right\| + \left\| (\dot{\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger})(\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger)^+ \right\| \right) \end{aligned} \tag{6.66}$$

with

$$\begin{aligned} \left\| \Lambda_p (\mathbf{I} - \mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger) \right\| &\leq \lambda_{max}(\Lambda_p)(1 + \|\mathbf{L}_e\| \left\| \widehat{\mathbf{L}}_e^\dagger \right\|) \\ \left\| (\dot{\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger})(\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger)^+ \right\| &\leq \left(\left\| \dot{\mathbf{L}_e} \right\| \left\| \widehat{\mathbf{L}}_e^\dagger \right\| + \|\mathbf{L}_e\| \left\| \dot{\widehat{\mathbf{L}}_e^\dagger} \right\| \right) \left\| (\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger)^+ \right\| \end{aligned} \tag{6.67}$$

In analyzing δA_P, if $\mathbf{L}_e$, $\widehat{\mathbf{L}}_e$ and $\dot{\mathbf{L}}_e$ are shown to be bounded, then the proof will be completed.

Boundedness of $\mathbf{L}_e$ and $\widehat{\mathbf{L}}_e$. The bounds of $\widehat{\mathbf{L}}_e^\dagger$ can be deduced from those of $\widehat{\mathbf{L}}_e$ given that

$$\sigma_{min}(\mathbf{L}_e) \leq \|\mathbf{L}_e\| \leq \sigma_{max}(\mathbf{L}_e) \tag{6.68}$$

then

$$\frac{1}{\sigma_{max}(\mathbf{L}_e)} \leq \left\| \mathbf{L}_e^\dagger \right\| \leq \frac{1}{\sigma_{min}(\mathbf{L}_e)} \tag{6.69}$$

The interaction matrix $\mathbf{L}_e$ is always associated with a given set of features, which depend on image measurements. In the definition of most geometrical features, the image point coordinates are used as basic components, with in addition the inverse value of depth [71, 36] or range (spherical projection based features) [73, 84, 85, 86] of the scene. For any image point $p(x, y)$, due to the finite dimension of the image sensor, there exist two constants $l_1 > 0$ and $l_2 > 0$ such that for all values of x and y belonging to the image, we have $|x| \leq l_1$ and $|y| \leq l_2$. Moreover the inverse value of the depth is bounded such that $0 < \frac{1}{Z} < \frac{1}{f}$, where f is the focal length. Hence, the boundedness of all image features components, implies that of $\mathbf{L}_e$.

However, the boundedness of $\mathbf{L}_e$ does not necessarily implies that of $\mathbf{L}_e^\dagger$, since a rank deficient $\mathbf{L}_e$ matrix has zero as minimum singular value ($\sigma_{min}(\mathbf{L}_e) = 0$). In such a case, the upper bound of $\mathbf{L}_e^\dagger$ may not exist. Therefore, a restriction to the case where $\mathbf{L}_e$ is of maximum rank is necessary to guarantee the boundedness of $\mathbf{L}_e^\dagger$.

Boundedness of $\dot{\mathbf{L}}_e$. Consider now the time derivative of $\mathbf{L}_e = \mathbf{L}_e(\xi, Z)$, which can be written as

$$\begin{aligned} \dot{\mathbf{L}}_e^T(\xi, Z) &= \dot{\xi}^T \frac{\partial \mathbf{L}_e}{\partial \xi}^T + \dot{Z}^T \frac{\partial \mathbf{L}_e}{\partial Z}^T \\ &= \begin{bmatrix} \dot{\xi}^T & \dot{Z}^T \end{bmatrix} \begin{bmatrix} \frac{\partial \boldsymbol{L}_e}{\partial \xi} & \frac{\partial \boldsymbol{L}_e}{\partial Z} \end{bmatrix}^T \end{aligned} \tag{6.70}$$

where $\dot{\xi} = \mathbf{L}_e {}^cV$ is the feature velocity vector. We can also define the Jacobian mapping $\dot{Z}_i$ to ${}^cV$ as $L_{Z_i} = Z_i[\ 0 \quad 0 \quad 1/Z_i \quad -y_i \quad x_i \quad 0\ ]$. Thus, we can write a general upper bound on $\dot{\mathbf{L}}_e(\xi, Z)$ as follows

$$\left\| \dot{\mathbf{L}}_e(\xi, Z) \right\| \leq (\|\mathbf{L}_e\| + \|L_Z\|)\, \|{}^cV\|\, \|\nabla_{\xi,Z} \mathbf{L}_e\| \tag{6.71}$$

In the particular case where ξ corresponds to feature points, we have

$$\dot{\mathbf{L}}_e(\xi, Z) = \begin{bmatrix} {}^cV^T \mathbf{H}_x \\ {}^cV^T \mathbf{H}_y \end{bmatrix} \tag{6.72}$$

where $\mathbf{H}_x$ and $\mathbf{H}_y$ are Hessian matrices [34, p. 73]. These matrices depend only on the image point coordinates and the inverse of the point depth, which have all been shown to be bounded. Then $\mathbf{H}_x$ and $\mathbf{H}_y$ are bounded such that $\|\mathbf{H}_x\| \leq \sigma_{max}(\mathbf{H}_x)$ and $\|\mathbf{H}_y\| \leq \sigma_{max}(\mathbf{H}_y)$. Hence, from (6.72) it can be shown that

$$\left\| \dot{\mathbf{L}}_e(\xi, Z) \right\| \leq \|{}^cV\| \sqrt{\sigma_{max}^2(\mathbf{H}_x) + \sigma_{max}^2(\mathbf{H}_y)} \tag{6.73}$$

The camera velocity being bounded, and considering Equations (6.66),(6.67), (6.68), (6.69) and (6.71) or (6.73) under the assumption that the visual servoing system is operating away from singular configurations, the perturbation $g(\boldsymbol{\epsilon}, t)$ can now be upper bounded as

$$\|g(\boldsymbol{\epsilon}, t)\| \leq \sigma_{max}(\delta A_P)\, \|\epsilon\| \tag{6.74}$$

Considering the set of Equations (6.56)-(6.60) and (6.62)-(6.74), by identification we have

$$\begin{cases} c_1 &= \lambda_{min}(P_N) \\ c_2 &= \lambda_{max}(P_N) \\ c_3 &= \lambda_{min}(Q_N) \\ c_4 &= 2\lambda_{max}(P_N) \\ \gamma &= \sigma_{max}(\delta A_P) \end{cases} \tag{6.75}$$

Then

$$\begin{aligned} \dot{\mathcal{L}}_N(\boldsymbol{\epsilon}, t) &\leq -c_3 \left\|\epsilon\right\|^2 + \left\| \frac{\partial \mathcal{L}_N}{\partial \boldsymbol{\epsilon}} \right\| \left\| g(\boldsymbol{\epsilon}, t) \right\| \\ &\leq -\lambda_{min}(Q_N) \left\|\epsilon\right\|^2 + 2\lambda_{max}(P_N)\sigma_{max}(\delta A_P) \left\|\epsilon\right\|^2 \end{aligned} \tag{6.76}$$

Therefore, we can conclude that the system is locally stable and will exponentially converge to the equilibrium point as long as

$$\sigma_{max}(\delta A_P) < \frac{\lambda_{min}(Q_N)}{2\lambda_{max}(P_N)} \tag{6.77}$$

Using the Lyapunov equation, the inequality $2\left\|A_N\right\|\left\|P_N\right\| \geq \left\|Q_N\right\|$ holds; this implies that $\left\|A_N\right\| \geq \frac{\left\|Q_N\right\|}{2\left\|P_N\right\|}$. Hence, the condition of convergence (6.77) can be related to the gains Λ_p and Λ_v through matrix A_N as follows

$$\sigma_{max}(\delta A_P) < \lambda_{min}(A_N) \tag{6.78}$$

Finally, from (6.62), (6.76) and (6.78), it can be shown the following exponential convergence

$$\mathcal{L}_N(\boldsymbol{\epsilon}, t) = \mathcal{L}_N(\boldsymbol{\epsilon}, 0) e^{-2(\lambda_{min}(A_N) - \sigma_{max}(\delta A_P))t} \tag{6.79}$$

Speed of Response

The proposed control law offers the additional benefit of speeding up the time response of the servoing system, while maintaining the same maximum velocity demand as the classical visual control law (2.14). In fact, the magnitude of velocity in the classical visual control law is given by

$$\left\|{}^cV\right\| = \left\| \widehat{\boldsymbol{L}}_e^{\dagger} \right\| \left\|\Lambda_p\right\| \left\|e(t)\right\| \tag{6.80}$$

while in the proposed visual control law

$$\left\|{}^cV\right\| = \left\| \widehat{\boldsymbol{L}}_e^{\dagger} \right\| \left\| \Lambda_v \left[sI + \Lambda_v\right]^{-1} \right\| \left\|\Lambda_p\right\| \left\|e(t)\right\| \tag{6.81}$$

where $\left\|\Lambda_v \left[sI + \Lambda_v\right]^{-1}\right\| \leq 1$ depending on the frequency. This means that for the same values of $\left\|\widehat{\boldsymbol{L}}_e^{\dagger}\right\|$, $\|\Lambda_p\|$ and $\|e(t)\|$ the possible maximum velocity of both laws are the same.

Consider now, for simplicity, the ideal closed loop equation with the classical exponential decrease ($\dot{\boldsymbol{e}}(t) = -\Lambda_p \boldsymbol{e}(t)$) and the closed loop Equation (6.10) ($\ddot{\boldsymbol{e}}(t) + \Lambda_v \dot{\boldsymbol{e}}(t) + \Lambda_v \Lambda_p \boldsymbol{e}(t) = 0$). From linear system theory, it can be shown that the settling times at 2% are respectively given by

$$t_{s2\%_{Classical}} \approx \frac{4}{\Lambda_{p_i}} \tag{6.82}$$

and

$$t_{s2\%_{Proposed}} \approx \frac{4}{1/2\Lambda_{v_i}} \tag{6.83}$$

Improving the settling time in (6.82) will automatically result in increasing the velocity demand in (6.80), while in equation (6.83), one can improve the system's speed without affecting the velocity demand (6.81). Thus, using the proposed law, two degrees of freedom are available to tune the speed and the damping of the servoing system, which of course depends on Λ_p.

6.4.4 Alternative Dynamic Visual Servoing Law

In implementing a dynamic compensation scheme, the visual servoing controller has to generate reference accelerations in addition to reference velocities. The simplest solution is to derive the reference velocity generated by the visual controller [189, 190, 193, 194, 196, 197, 192, 188]. However, when using the classical law (2.14)), this solution implies a direct time-derivation of image features, which inevitably amplifies noise. In the proposed law however, thanks to the filter, no direct derivative of the image measurements will be needed. Hence, from the reference velocity given by

$${}^{c}V_r = -\widehat{\mathbf{L}}_e^{\dagger} \Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\Lambda_p \boldsymbol{e}(t)\right) \tag{6.84}$$

multiplying both sides of (6.84) by $\left[s\mathbf{I} + \Lambda_v\right] \Lambda_v^{-1} \widehat{\mathbf{L}}_e$ and simplifying gives the acceleration as follows

$${}^{c}\dot{V}_r = -\Lambda_v \Lambda_p \boldsymbol{e}(t) - \left(\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger} \dot{\widehat{\mathbf{L}}}_e\right) {}^{c}V_r \tag{6.85}$$

Each term in Equation (6.85) is known, except $\dot{\widehat{\mathbf{L}}}_e$ which has to be estimated. For this reason, one may think that the advantage of the proposed law could be partially lost. However, $\dot{\widehat{\mathbf{L}}}_e$ could determined analytically; for instance, when image features are considered, as shown previously (6.72), $\dot{\widehat{\mathbf{L}}}_e$ will be given by

$$\dot{\widehat{\mathbf{L}}}_e = \begin{bmatrix} {}^cV_r^T \mathbf{H}_x \\ {}^cV_r^T \mathbf{H}_y \end{bmatrix}$$

Otherwise, the approximate matrix $\widehat{\mathbf{L}}_e$ could be chosen such that its derivative $\dot{\widehat{\mathbf{L}}}_e$ be obtained easily if not by time-derivative. Moreover, even if the time-derivative is adopted, the product $\widehat{\mathbf{L}}_e^\dagger \dot{\widehat{\mathbf{L}}}_e$ in (6.85) is much smaller when compared to Λ_v, as we will see in simulations.

Nevertheless, we propose an alternative dynamic law, which does not require any time-derivative of $\widehat{\mathbf{L}}_e$ and where the reference velocity is computed as follows

$${}^cV_r = -\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\widehat{\mathbf{L}}_e^\dagger \Lambda_p \boldsymbol{e}(t)\right) \tag{6.86}$$

In this law, the filter is applied to the product $\widehat{\mathbf{L}}_e^\dagger \Lambda_p \boldsymbol{e}(t)$ rather than $\Lambda_p \boldsymbol{e}(t)$ alone as in (6.84). Thus, following the same procedure as above, the reference acceleration is obtained as

$${}^c\dot{V}_r = -\Lambda_v \widehat{\mathbf{L}}_e^\dagger \Lambda_p \boldsymbol{e}(t) - \Lambda_v {}^cV_r \tag{6.87}$$

Note that all terms in Equation (6.87) are known; ${}^c\dot{V}_r$ can therefore be computed from image measurements only, with no need to compute the derivative of any quantity.

To determine the closed loop equation resulting from law (6.86), let us substitute (6.86) in the open loop equation $\dot{\boldsymbol{e}}(t) = \mathbf{L}_e {}^cV$. We have

$$\dot{\boldsymbol{e}}(t) = -\mathbf{L}_e \Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\widehat{\mathbf{L}}_e^\dagger \Lambda_p \boldsymbol{e}(t)\right) \tag{6.88}$$

Assuming that $\mathbf{L}_e^\dagger \mathbf{L}_e = \mathbf{I}$, after multiplying both sides of Equation (6.88) by $\left[s\mathbf{I} + \Lambda_v\right] \mathbf{L}_e^\dagger$ and simplifying, we obtain

$$\mathbf{L}_e^\dagger \ddot{\boldsymbol{e}}(t) + \dot{\mathbf{L}}_e^\dagger \dot{\boldsymbol{e}}(t) + \Lambda_v \mathbf{L}_e^\dagger \dot{\boldsymbol{e}}(t) = -\Lambda_v \widehat{\mathbf{L}}_e^\dagger \Lambda_p \boldsymbol{e}(t) \tag{6.89}$$

with $\dot{\mathbf{L}}_e^\dagger = -\mathbf{L}_e^\dagger \dot{\mathbf{L}}_e \mathbf{L}_e^\dagger$. Multiplying again both sides of (6.89) by $\mathbf{L}_e$ and grouping finally yields

$$\ddot{\boldsymbol{e}}(t) + \left(\Lambda_v - \dot{\mathbf{L}}_e \mathbf{L}_e^\dagger\right) \dot{\boldsymbol{e}}(t) + \Lambda_v \mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger \Lambda_p \boldsymbol{e}(t) = 0 \tag{6.90}$$

The stability analysis of (6.90) can be carried out similarly to that of (6.45). The closed loop (6.90) is stable provided that $\left(\Lambda_v - \dot{\mathbf{L}}_e \mathbf{L}_e^\dagger\right) > 0$ and $\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger > 0$.

Note that because of the term $\dot{\mathbf{L}}_e \mathbf{L}_e^\dagger$ in (6.90), even if $\mathbf{L}_e$ is perfectly approximated such that $\mathbf{L}_e \widehat{\mathbf{L}}_e^\dagger = \mathbf{I}$, the closed loop would not be equal to the ideal one (6.10).

6.5 Dynamic Visual Servo Tracking

In the previous section, we have derived the closed loop equation resulting from the proposed control law (6.40) under the assumption that the target object was motionless. We are now interested in the case where both the camera and the target object are allowed to move. In this case, $\frac{\partial \boldsymbol{e}}{\partial t} = -\mathbf{L}_e {}^cV_o \neq 0$ where ${}^cV_o$ denotes the target velocity expressed in the camera frame. The task function is then given by

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_e {}^cV - \mathbf{L}_e {}^cV_o \tag{6.91}$$

6.5.1 Closed loop Dynamics

Considering a moving target, the closed loop dynamics under control law (6.40) can be derived by following the same procedure as in the motionless object's case in section 6.4.2. Hence, the following dynamics is obtained

$$\begin{cases} \ddot{\boldsymbol{e}}(t) + \left(\Lambda_v - (\mathbf{L}_e \dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^{+}\right) \dot{\boldsymbol{e}}(t) + \Lambda_v \Lambda_p (\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger}) \boldsymbol{e}(t) = \boldsymbol{\Psi} \\ \boldsymbol{\Psi} = -\mathbf{L}_e ({}^c\dot{V}_0 + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger} \dot{\widehat{\mathbf{L}}}_e) {}^cV_0) \end{cases} \tag{6.92}$$

The left-hand side of the first row of Equation (6.92) is exactly the task function dynamics due to the camera when the object is motionless, whereas the right-hand side $\boldsymbol{\Psi}$, whose expression is given in the second row represents the dynamics of the task function due to the object motion.

Through the term $\mathbf{L}_e {}^c\dot{V}_0$ in the expression of $\boldsymbol{\Psi}$, we can see that using the proposed control law (6.40) results in a closed loop Equation (6.92) accounting automatically for a possible accelerating target. This is an interesting feature that could be exploited in the compensation of the object motion.

6.5.2 Convergence Analysis

To analyze the convergence of the task function in the presence of a moving object, let us start by writing Equation (6.92) in state space notation. Hence, we have

$$\frac{d}{dt}\boldsymbol{\epsilon} = A_S \boldsymbol{\epsilon} + B_S \boldsymbol{\Psi} \tag{6.93}$$

with

$$A_S = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\Lambda_v \Lambda_p (\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger}) & -(\Lambda_v - (\mathbf{L}_e \dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e \widehat{\mathbf{L}}_e^{\dagger})^{+}) \end{bmatrix} \text{ and } B_S = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix}$$

Under the assumptions given in section (6.4.3), the autonomous system $\dot{\boldsymbol{\epsilon}} = A_S \boldsymbol{\epsilon}$ was shown to be stable. Using the same assumptions, the system (6.93) is controllable, since it can be shown that

$$det\begin{bmatrix} B_S & A_S B_S \end{bmatrix} = \begin{vmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & -(\Lambda_v - (\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+}) \end{vmatrix} \neq 0 \tag{6.94}$$

Therefore, according to (6.93), the system's output $\boldsymbol{e}(t)$ will not tend asymptotically towards zero, but instead towards a value function of the input $\boldsymbol{\Psi}$ and the system's gain. A compensation is then required since the servoing will fail to reach the relative desired pose when the target is moving.

6.5.3 Object's Motion Compensation

The object's motion, from the control point of view, can be seen as an input disturbance acting on the system. Thus, if constant, the object's velocity could be compensated by a simple integral action in the control law [162]. However, if the object's velocity is not constant, which is more likely to happen in practice, the simple integral action would not solve the problem. A better solution could be to use a feed-forward controller if an estimate of the disturbance is available [203], [163], [26], [68].

Adopting the last solution, with the proposed control laws (6.40) and (6.86), there are two ways to add a feed-forward action. It could be either filtered together with $\boldsymbol{e}(t)$ or added without being filtered. In the last case, the velocity command will be given by

$${}^{c}V_r = -\widehat{\mathbf{L}}_e^{\dagger}(\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \Lambda_p \boldsymbol{e}(t)) + {}^{c}\widehat{V}_{o_{ff}} \tag{6.95}$$

where ${}^{c}\widehat{V}_{o_{ff}}$ represents the estimate value of the object's velocity. Following the same procedure as previously, the resulting closed loop equation is given by

$$\begin{cases} \ddot{\boldsymbol{e}}(t) + \left(\Lambda_v - (\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+}\right)\dot{\boldsymbol{e}}(t) + \Lambda_v\Lambda_p(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})\boldsymbol{e}(t) = \boldsymbol{\Psi} \\ \boldsymbol{\Psi} = \mathbf{L}_e({}^{c}\dot{\widehat{V}}_{o_{ff}} + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^{c}\widehat{V}_{o_{ff}}) - \mathbf{L}_e({}^{c}\dot{V}_0 + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^{c}V_0) \end{cases} \tag{6.96}$$

Equation (6.96) shows clearly that if ${}^{c}\widehat{V}_{o_{ff}}$ is a good estimate of ${}^{c}V_0$, the expression of $\boldsymbol{\Psi}$ would be negligible and the closed loop will behave as the undisturbed system (6.45).

When the feed-forward action is added through the filter, the velocity command will be given by

$${}^{c}V_r = -\widehat{\mathbf{L}}_e^{\dagger}(\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} (\Lambda_p \boldsymbol{e}(t)) - \widehat{\mathbf{L}}_e{}^{c}\widehat{V}_{o_{ff}}) \tag{6.97}$$

where the product $\widehat{\mathbf{L}}_e{}^{c}\widehat{V}_{o_{ff}}$ has to be seen as the estimate value of the effect of object's motions on $\dot{\boldsymbol{e}}(t)$. It can be shown that the resulting closed loop equation will be given by

$$\begin{cases} \ddot{\boldsymbol{e}}(t) + \left(\Lambda_v - (\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+}\right)\dot{\boldsymbol{e}}(t) + \Lambda_v\Lambda_p(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})\boldsymbol{e}(t) = \boldsymbol{\Psi} \\ \boldsymbol{\Psi} = \Lambda_v\mathbf{L}_e{}^{c}\widehat{V}_{o_{ff}} - \mathbf{L}_e({}^{c}\dot{V}_0 + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^{c}V_0) \end{cases} \tag{6.98}$$

To compensate the object motion, From Equation (6.98), it can be seen that the object motion can be compensated without the feed-forward input ${}^c\widehat{V}_{o_{ff}}$ being equal to the object velocity ${}^cV_0$. We comment next on the two compensation cases of the target's motion, namely when the velocity is constant, and when it is varying.

Case 1: Target with Constant Velocity

When the target velocity is constant, the acceleration ${}^c\dot{V}_0$ is zero. The input $\mathbf{\Psi}$ to the system (6.98) becomes

$$\mathbf{\Psi} = \Lambda_v \mathbf{L}_e {}^c\widehat{V}_{o_{ff}} - \mathbf{L}_e(\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^cV_0 \tag{6.99}$$

To compensate for the object's motion, ${}^c\widehat{V}_{o_{ff}}$ has to be designed such that

$${}^c\widehat{V}_{o_{ff}} = (\mathbf{I} + \Lambda_v^{-1}\widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^c\widehat{V}_0 \tag{6.100}$$

Clearly, Equation (6.100) shows that with a good estimate ${}^c\widehat{V}_o$ of the object's velocity, the value of the input $\mathbf{\Psi}$ in the closed loop system will almost be nullified. Hence, allowing the task function to converge towards zero.

Case 2: Accelerating Target

Unlike the previous case, in order to cancel the effects of both the target's velocity and acceleration, ${}^c\widehat{V}_{o_{ff}}$ has to be designed such that

$${}^c\widehat{V}_{o_{ff}} = \Lambda_v^{-1}{}^c\dot{\widehat{V}}_o + (\mathbf{I} + \Lambda_v^{-1}\widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^c\widehat{V}_0 \tag{6.101}$$

Remark. The feed-forward input $\widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}$ should be regarded as an additional control variable in the design process, which can be used not only to cancel the effects of disturbances related to object's motion, but also to make the system robust with respect to modeling errors or other uncertainties. It can be designed, for example, as a variable structure input [187] to compensate for interactions between the vision system and the robot dynamics, as we will see shortly.

6.6 Dynamic Visual Servo Compensation

In deriving the closed loop equation, it was implicitly assumed that the robot executes perfectly the reference velocity. According to that assumption, the robot was seen as a kinematic device, ignoring as such its dynamics. In this section, we relax that assumption and consider the full nonlinear robot dynamics. We start by modeling how the latter affects the visual tasks and afterward we design a compensation scheme.

6.6.1 Camera-Robot: Dynamic Interaction

Consider the task function (6.91) and the reference velocity (6.95), which are recalled here

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_e {}^cV - \mathbf{L}_e {}^cV_o \tag{6.102}$$

and

$${}^cV_r = -\widehat{\mathbf{L}}_e^{\dagger}\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\Lambda_p e(t) - \widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}\right) \tag{6.103}$$

Let the error between the reference velocity ${}^cV_r$ and the actual velocity of the camera ${}^cV$ be defined as

$${}^c\widetilde{V} \triangleq {}^cV_r - {}^cV \tag{6.104}$$

From (6.104) and the expression of ${}^cV_r$, the velocity ${}^cV$ can be written as

$${}^cV = -\widehat{\mathbf{L}}_e^{\dagger}\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\Lambda_p e(t) - \widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}\right) - {}^c\widetilde{V} \tag{6.105}$$

Substituting Equation (6.105) in the task function (6.102) gives

$$\dot{\boldsymbol{e}}(t) = -\mathbf{L}_e \left(\widehat{\mathbf{L}}_e^{\dagger}\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\Lambda_p e(t) - \widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}\right) + {}^c\widetilde{V}\right) - \mathbf{L}_e {}^cV_o \tag{6.106}$$

After developing and simplifying (6.106), we obtain

$$\begin{cases} \ddot{\boldsymbol{e}}(t) + \left(\Lambda_v - (\mathbf{L}_e\dot{\widehat{\mathbf{L}}}_e^{\dagger})(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})^{+}\right)\dot{\boldsymbol{e}}(t) + \Lambda_v\Lambda_p(\mathbf{L}_e\widehat{\mathbf{L}}_e^{\dagger})\boldsymbol{e}(t) = \boldsymbol{\Psi} \\ \boldsymbol{\Psi} = \Lambda_v\mathbf{L}_e {}^c\widehat{V}_{o_{ff}} - \mathbf{L}_e({}^c\dot{V}_0 + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^cV_0) \\ \qquad -\mathbf{L}_e({}^c\dot{\widetilde{V}} + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^c\widetilde{V}) \end{cases} \tag{6.107}$$

Under the proposed control law (6.95), Equation (6.106) tells us that the robot's dynamics will affect, through the term $\mathbf{L}_e({}^c\dot{\widetilde{V}} + (\Lambda_v + \widehat{\mathbf{L}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^c\widetilde{V})$, the dynamics of the task function $\boldsymbol{e}(t)$. Obviously, if the robot has poor tracking performances, the task function will not decrease to zero in a tracking application. As a result, the visual servoing system will fail to reach the desired relative pose and will therefore need a compensation aiming at solving this problem.

Often the robot is controlled in joint space, it is suitable to rewrite the closed loop equation in joint space. Thus, starting from the task function which becomes

$$\dot{\boldsymbol{e}}(t) = \mathbf{J}_e\dot{\mathbf{q}} - \mathbf{L}_e {}^cV_o \tag{6.108}$$

where $\mathbf{J}_e$ is the Jacobian of the visual task given by $\mathbf{J}_e = \mathbf{L}_e {}^c\mathbf{J}_q$, with ${}^c\mathbf{J}_q$ being the robot Jacobian expressed in the camera frame. The reference joints velocity and acceleration vectors can be shown to be respectively given by

$$\dot{\mathbf{q}}_r = -\widehat{\mathbf{J}}_e^{\dagger}\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\Lambda_p \boldsymbol{e}(t) - \widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}\right) \tag{6.109}$$

and

$$\ddot{\mathbf{q}}_r = -\Lambda_v \widehat{\mathbf{J}}_e^{\dagger}(\Lambda_p \boldsymbol{e}(t) - \widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}) - (\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\dot{\mathbf{q}}_r \tag{6.110}$$

Similarly, by defining $\dot{\tilde{\mathbf{q}}} \triangleq \dot{\mathbf{q}}_r - \dot{\mathbf{q}}$ as the joints velocity error, the actual robot joints velocity vector $\dot{\mathbf{q}}$ will be given by

$$\dot{\mathbf{q}} = -\widehat{\mathbf{J}}_e^{\dagger}\Lambda_v \left[s\mathbf{I} + \Lambda_v\right]^{-1} \left(\Lambda_p \boldsymbol{e}(t) - \widehat{\mathbf{L}}_e {}^c\widehat{V}_{o_{ff}}\right) - \dot{\tilde{\mathbf{q}}} \tag{6.111}$$

From the substitution of (6.108) in (6.111) and following the same procedure as in Section 6.4.2 with the assumption that ${}^c\mathbf{J}_q$ can accurately be estimated, we obtain the closed loop dynamics of the task function as follows

$$\begin{cases} \ddot{\boldsymbol{e}}(t) + (\Lambda_v - (\mathbf{J}_e\dot{\widehat{\mathbf{J}}}_e^{\dagger})(\mathbf{J}_e\widehat{\mathbf{J}}_e^{\dagger})^{+})\dot{\boldsymbol{e}}(t) + \Lambda_v(\mathbf{J}_e\widehat{\mathbf{J}}_e^{\dagger})\Lambda_p\boldsymbol{e}(t) = \boldsymbol{\Psi} \\ \boldsymbol{\Psi} = \Lambda_v\mathbf{L}_e{}^c\widehat{V}_{o_{ff}} - \mathbf{L}_e({}^c\dot{V}_0 + (\Lambda_v + {}^c\mathbf{J}_q\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e){}^cV_0) \\ \qquad -\mathbf{J}_e(\ddot{\tilde{\mathbf{q}}} + (\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\dot{\tilde{\mathbf{q}}}) \end{cases} \tag{6.112}$$

From (6.112), it can seen that the term $\mathbf{J}_e(\ddot{\tilde{\mathbf{q}}} + (\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\dot{\tilde{\mathbf{q}}})$ represents the error dynamics in the task function due to tracking error in the joint space. This term reflects the robot dynamics into the visual space.

The robot dynamics on the other hand could be written as

$$\begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{M}^{-1}(\mathbf{q})(\mathbf{N}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) + \Phi(\mathbf{q},\dot{\mathbf{q}})) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1}(\mathbf{q}) \end{bmatrix} \boldsymbol{u} \tag{6.113}$$

where $\mathbf{M}(\mathbf{q}) = \mathbf{D}(\mathbf{q}) + \mathbf{J}$, $\mathbf{N}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{B}$, $G(\mathbf{q})$ and $\Phi(\mathbf{q},\dot{\mathbf{q}}) = F(\dot{\mathbf{q}}) + T_d$ are described in Section (6.3). In case the robot is already controlled, a general form of control input could be written as [36, 187]

$$\boldsymbol{u} = \widehat{\mathbf{M}}(\mathbf{q})\ddot{\hat{\mathbf{q}}}_{ref} + \widehat{\mathbf{N}}(\mathbf{q},\dot{\mathbf{q}})\dot{\hat{\mathbf{q}}}_{ref} + G(\mathbf{q}) + \widehat{\Phi} \tag{6.114}$$

where $\mathbf{q}_{ref}$ denotes the reference joint variables.

Equations (6.112) and (6.113) represent the dynamic model of the visual servoing system. It consists of two cascaded subsystems, the visual servo controller and the robot. The overall system has to be compensated such that the robot accurately follows the reference trajectories generated by the visual controller.

6.6.2 Camera-Robot: Dynamic Compensation

Prior the design of the controller, let us recall some properties of the robot's dynamic model [126, pp. 216 - 219] that will be used in the control design.

Property 1: The inertia matrix $\mathbf{D}(\mathbf{q})$ is symmetric, positive-definite and bounded such that

$$\alpha_1 \|\boldsymbol{x}\|^2 \leq \boldsymbol{x}^T \mathbf{D}(\mathbf{q}) \boldsymbol{x} \leq \alpha_2 \|\boldsymbol{x}\|^2 \quad \forall \boldsymbol{x} \in \mathbb{R}^n$$

where α_1 , $\alpha_2 \in \mathbb{R}_+$ are constants defining the bounds of $\mathbf{D}(\mathbf{q})$.
Property 2 : The matrix $(\frac{1}{2}\dot{\mathbf{D}}(\mathbf{q}) - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}))$ is skew-symmetric such that

$$\boldsymbol{x}^T (\frac{1}{2}\dot{\mathbf{D}}(\mathbf{q}) - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}))\boldsymbol{x} = 0 \quad \forall \mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{x} \in \mathbb{R}^n$$

Property 3: The robot dynamic model (6.113) can be expressed linearly as a function of system's *parameter vector* $\boldsymbol{\theta}_d \in \mathbb{R}^m$ such that

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) + \Phi(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{Y}_d(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\boldsymbol{\theta}_d$$

where $\overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) \in \mathbb{R}^{n \times m}$ is called dynamic *regressor* containing known functions whose variables are $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$.

The dynamic compensation of the two cascaded subsystems can be done differently. Essentially our design will be based on robust adaptive schemes. On one hand, the robust compensation is meant to cope with uncertainties, unmodeled dynamics resulting from many assumptions, and of course to deal with disturbances [189, 190, 196]. On the other hand, the adaptive compensation aims at improving the estimated values. This could result in an improvement of the control scheme, especially on the robot side [193]. Hence, as illustrated in Figure 6.1, we will use in conjunction with the visual control law (6.112) the following controller

$$\begin{aligned} \boldsymbol{u} &= \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\dot{\mathbf{q}}}_r)\widehat{\boldsymbol{\theta}}_d + \widehat{T}_d + \mu \\ &= \widehat{\mathbf{M}}(\mathbf{q})\hat{\dot{\mathbf{q}}}_r + \widehat{\mathbf{N}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_r + G(\mathbf{q}) + \widehat{\Phi} + \boldsymbol{\mu} \end{aligned} \tag{6.115}$$

where μ is an auxiliary control signal to be determined during the design, $\overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\dot{\mathbf{q}}}_r)$ denotes the dynamic regressor of the robot written as function of the modified reference trajectories $(\dot{\mathbf{q}}_r, \hat{\dot{\mathbf{q}}}_r)$. The modified reference signal $\hat{\dot{\mathbf{q}}}_r$ is defined as follows

$$\hat{\dot{\mathbf{q}}}_r \triangleq \ddot{\mathbf{q}}_r - \alpha_v (\dot{\mathbf{q}} - \dot{\mathbf{q}}_r) \tag{6.116}$$

where α_v is positive constant and is an additional design parameter to be tuned.

The control law $\overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\dot{\mathbf{q}}}_r)\widehat{\boldsymbol{\theta}}_d$ with an updating law for the estimated parameters $\widehat{\boldsymbol{\theta}}_d$ was

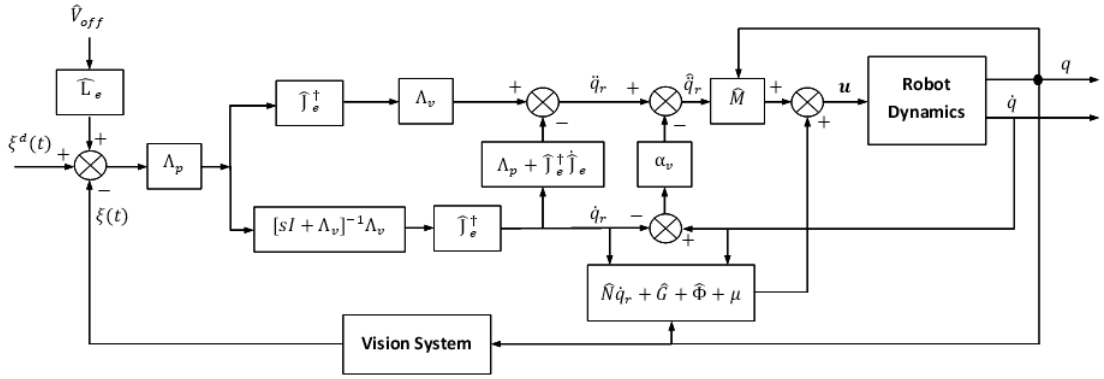


Figure 6.1: Compensation scheme of dynamic visual servoing

first proposed in [201, p. 405], where the global asymptotic stability and tracking convergence were proved. This law has been widely used in robotic for tracking applications [126, 186, 187].

6.6.2.1 Controller Design: Formulation

Let us define the following sliding variables [201, p. 399]

$$\begin{aligned} \boldsymbol{\nu} &\triangleq \dot{\mathbf{q}} - \dot{\mathbf{q}}_r \\ \dot{\boldsymbol{\nu}} &\triangleq \ddot{\mathbf{q}} - \ddot{\mathbf{q}}_r \end{aligned} \tag{6.117}$$

Using definition (6.116) and Equation (6.117), the dynamic of the robot can be written as function of $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$ such that the equilibrium point in closed loop be moved to the origin. First of all we have

$$\begin{aligned} \dot{\mathbf{q}} &= \boldsymbol{\nu} + \dot{\mathbf{q}}_r \\ \ddot{\mathbf{q}} &= \dot{\boldsymbol{\nu}} + \alpha_v \boldsymbol{\nu} + \hat{\ddot{\mathbf{q}}}_r \end{aligned} \tag{6.118}$$

Substituting (6.118) in (6.113) and dropping all dependencies, it can be shown that

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{N}\boldsymbol{\nu} + \alpha_v \mathbf{M}\boldsymbol{\nu} + \mathbf{M}\hat{\ddot{\mathbf{q}}}_r + \mathbf{N}\dot{\mathbf{q}}_r + G + \Phi = \boldsymbol{u} \tag{6.119}$$

where the last four terms are nothing but $\overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\ddot{\mathbf{q}}}_r)\boldsymbol{\theta}_d$. Hence, the robot's model (6.119) can now be written as

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{N}\boldsymbol{\nu} + \alpha_v \mathbf{M}\boldsymbol{\nu} + \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\ddot{\mathbf{q}}}_r)\boldsymbol{\theta}_d = \boldsymbol{u} \tag{6.120}$$

Substituting the control law (6.115) in (6.120), it can be shown that the closed loop equation of the robot subsystem will be given by

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{N}\boldsymbol{\nu} + \alpha_v \mathbf{M}\boldsymbol{\nu} = \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\ddot{\mathbf{q}}}_r)(\widehat{\boldsymbol{\theta}}_d - \boldsymbol{\theta}_d) + \boldsymbol{\mu} \tag{6.121}$$

By defining $\widetilde{\boldsymbol{\theta}}_d \triangleq \widehat{\boldsymbol{\theta}}_d - \boldsymbol{\theta}_d$ as the error in estimated parameters vector of the robot, Equation (6.121) becomes

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{N}\boldsymbol{\nu} + \alpha_v \mathbf{M}\boldsymbol{\nu} = \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \dot{\hat{\mathbf{q}}}_r)\widetilde{\boldsymbol{\theta}}_d + \boldsymbol{\mu} \tag{6.122}$$

At this stage, if for instance, we ignore $\boldsymbol{\mu}$ and add an updating law of the form

$$\dot{\widehat{\boldsymbol{\theta}}}_d = f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}) \tag{6.123}$$

to improve the estimated dynamic parameters $\widehat{\boldsymbol{\theta}}_d$, such that the time-derivative of a candidate Lyapunov function of the system (6.122) be negative-definite, we would obtain an adaptive control scheme [201, pp. 403 - 406]. If instead, despite the mismatch between $\widehat{\boldsymbol{\theta}}_d$ and $\boldsymbol{\theta}_d$, $\boldsymbol{\mu}$ is design to force the time-derivative of the mentioned Lyapunov function to be negative-definite, we would then obtain a robust control scheme [126, pp. 260 - 263].

Based on these considerations, our robust adaptive control problem of the overall system can be formulated by rewriting (6.112) and (6.122) and augmenting them with the adaptive law (6.123). Hence, we obtain

$$\begin{cases} \dot{\boldsymbol{\epsilon}} & = A_{S'}\boldsymbol{\epsilon} + B_{S'}\left\{\delta_{ac} - \eta_o + \eta_d\right\} \\ \dot{\boldsymbol{\nu}} & = -(\mathbf{M}^{-1}\mathbf{N} + \alpha_v \mathbf{I})\boldsymbol{\nu} + \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \dot{\hat{\mathbf{q}}}_r)\widetilde{\boldsymbol{\theta}}_d + \boldsymbol{\mu} \\ \dot{\widetilde{\boldsymbol{\theta}}}_d & = f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}, \boldsymbol{\epsilon}) \end{cases} \tag{6.124}$$

where $A_{S'}$ is similar to A_S given in section 6.5.2, but with $\mathbf{J}_e$ and $\mathbf{J}_e^\dagger$ in lieu of $\mathbf{L}_e$ and $\mathbf{L}_e^\dagger$, respectively; we also have the following definitions

$$\begin{aligned} \delta_{ac} &\triangleq \Lambda_v \mathbf{L}_e {}^c\widehat{V}_{o_{ff}} \\ \eta_o &\triangleq \mathbf{L}_e({}^c\dot{V}_0 + (\Lambda_v + {}^c\mathbf{J}_q \widehat{\mathbf{J}}_e^\dagger \dot{\widehat{\mathbf{L}}}_e){}^cV_0) \\ \eta_d &\triangleq \mathbf{J}_e(\dot{\boldsymbol{\nu}} + (\Lambda_v + \widehat{\mathbf{J}}_e^\dagger \dot{\widehat{\mathbf{J}}}_e)\boldsymbol{\nu}) \end{aligned} \tag{6.125}$$

Note that if a perfect convergence is achieved, all error signals in the overall system ($\boldsymbol{\epsilon} = [\ \boldsymbol{e}^T(t) \quad \dot{\boldsymbol{e}}^T(t)\ ]^T$, $\boldsymbol{\nu}$, $\dot{\boldsymbol{\nu}}$ and $\widetilde{\boldsymbol{\theta}}_d$) should be equal to zero.

6.6.2.2 Controller Design: Solution

The overall system dynamics, as formulated in (6.124), has its equilibrium at zero. Thus, Lyapunov stability and convergence theory [201, chap. 3] can be used to solve our controller design problem. Hence, this problem reduces in finding $\boldsymbol{\mu}$, $f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}, \boldsymbol{\epsilon})$ and ${}^c\widehat{V}_{o_{ff}}$ such that the time-derivative of a candidate Lyapunov function defined for the visual servoing system (6.124) be negative-definite.

To that end, we will derive the controller in the following steps:

Step 1. Candidate Lyapunov Function

Consider the following candidate Lyapunov function

$$\mathcal{L}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d) = \frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} \boldsymbol{\epsilon} + \frac{1}{2}\boldsymbol{\nu}^T \mathbf{M} \boldsymbol{\nu} + \frac{1}{2}\widetilde{\boldsymbol{\theta}}_d^T K_d^{-1} \widetilde{\boldsymbol{\theta}}_d \tag{6.126}$$

where $P_{S'}$ is a positive-definite matrix under the assumptions given in Section 6.4.3, such that $A_{S'}^T P_{S'} + P_{S'} A_{S'} = -Q_{S'}$ and K_d is a symmetric positive-definite matrix. Using property 1, it can be shown that $\mathcal{L}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d)$ is also a positive-definite function such that

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d) &\geq \tfrac{\lambda_{min}}{2}(P_{S'}) \left\|\boldsymbol{\epsilon}\right\|^2 + \tfrac{\alpha_1}{2} \left\|\boldsymbol{\nu}\right\|^2 + \tfrac{\lambda_{min}}{2}(K_d^{-1}) \left\|\widetilde{\boldsymbol{\theta}}_d\right\|^2 \\
\mathcal{L}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d) &\leq \tfrac{\lambda_{max}}{2}(P_{S'}) \left\|\boldsymbol{\epsilon}\right\|^2 + \tfrac{\alpha_2}{2} \left\|\boldsymbol{\nu}\right\|^2 + \tfrac{\lambda_{max}}{2}(K_d^{-1}) \left\|\widetilde{\boldsymbol{\theta}}_d\right\|^2
\end{aligned}$$

Step 2. Time Derivative of the Candidate Lyapunov Function

The time derivative of $\mathcal{L}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d)$ is given by

$$\dot{\mathcal{L}}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d) = \frac{1}{2}(\boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} \dot{\boldsymbol{\epsilon}} + \dot{\boldsymbol{\epsilon}}^{\boldsymbol{T}} P_{S'} \boldsymbol{\epsilon}) + \boldsymbol{\nu}^T \mathbf{M} \dot{\boldsymbol{\nu}} + \frac{1}{2}\boldsymbol{\nu}^T \dot{\mathbf{M}} \boldsymbol{\nu} + \widetilde{\boldsymbol{\theta}}_d^T K_d^{-1} \dot{\widetilde{\boldsymbol{\theta}}}_d \tag{6.127}$$

Substituting (6.124) in (6.127), and using definitions $\mathbf{M} = \mathbf{D} + \mathbf{J}$, $\mathbf{N} = \mathbf{C} + \mathbf{B}$ in conjunction with property 2 ($\boldsymbol{\nu}^T(\frac{1}{2}\dot{\mathbf{D}} - \mathbf{C})\boldsymbol{\nu} = 0$) gives

$$\begin{aligned}
\dot{\mathcal{L}}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d) = \frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}}(P_{S'} A_{S'} + A_{S'}^T P_{S'})\boldsymbol{\epsilon} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}(\delta_{ac} - \eta_o + \eta_d) \\
- \boldsymbol{\nu}^T(\mathbf{C} + \mathbf{B} + \alpha_v \mathbf{M})\boldsymbol{\nu} + \boldsymbol{\nu}^T \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\dot{\mathbf{q}}}_r)\widetilde{\boldsymbol{\theta}}_d + \boldsymbol{\nu}^T \boldsymbol{\mu} \\
\boldsymbol{\nu}^T \mathbf{C} \boldsymbol{\nu} + \widetilde{\boldsymbol{\theta}}_d^T K_d^{-1} f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}, \boldsymbol{\epsilon})
\end{aligned} \tag{6.128}$$

After simplifying and substituting $\eta_d = \mathbf{J}_e(\dot{\boldsymbol{\nu}} + (\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\boldsymbol{\nu})$, (6.128) becomes

$$\begin{aligned}
\dot{\mathcal{L}}(\boldsymbol{\epsilon}, \boldsymbol{\nu}, \widetilde{\boldsymbol{\theta}}_d) = -\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}} Q_{S'} \boldsymbol{\epsilon} - \boldsymbol{\nu}^T(\mathbf{B} + \alpha_v \mathbf{M})\boldsymbol{\nu} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}(\delta_{ac} - \eta_o) \\
+ \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e \dot{\boldsymbol{\nu}} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\boldsymbol{\nu} + \boldsymbol{\nu}^T \boldsymbol{\mu} \\
+ \boldsymbol{\nu}^T \overline{\mathbf{Y}}_d(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\dot{\mathbf{q}}}_r)\widetilde{\boldsymbol{\theta}}_d + \widetilde{\boldsymbol{\theta}}_d^T K_d^{-1} f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}, \boldsymbol{\epsilon})
\end{aligned} \tag{6.129}$$

Step 3. Choice of the Adaptive Law

Under the assumption that the dynamic parameters $\boldsymbol{\theta}_d$ are constant or slowly varying, we have

$$\dot{\widehat{\boldsymbol{\theta}}}_d \approx \dot{\widetilde{\boldsymbol{\theta}}}_d = f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}, \boldsymbol{\epsilon}) \tag{6.130}$$

As in [193, 194], we chose the following adaptive law

$$f(\overline{\mathbf{Y}}_d, \boldsymbol{\nu}, \boldsymbol{\epsilon}) = -K_d \overline{\mathbf{Y}}_d^T(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \hat{\ddot{\mathbf{q}}}_r)\boldsymbol{\nu} \tag{6.131}$$

Substituting (6.131) in (6.129) and dropping the variables of $\mathcal{L}$ to simplify notation, results in

$$\begin{aligned}\dot{\mathcal{L}} = &-\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}} Q_{S'}\boldsymbol{\epsilon} - \boldsymbol{\nu}^T(\mathbf{B} + \alpha_v \mathbf{M})\boldsymbol{\nu} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}(\delta_{ac} - \eta_o) \\ &+ \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e \dot{\boldsymbol{\nu}} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e (\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\boldsymbol{\nu} + \boldsymbol{\nu}^T \boldsymbol{\mu}\end{aligned} \tag{6.132}$$

Step 4. Design of the Additional Control Input $\boldsymbol{\mu}$

Consider Equation (6.132), if $\boldsymbol{\mu}$ is chosen such that

$$\boldsymbol{\mu} = -(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)^T \mathbf{J}_e^T B_{S'}^T P_{S'}\boldsymbol{\epsilon} \tag{6.133}$$

it will simplify in

$$\begin{aligned}\dot{\mathcal{L}} = &-\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}} Q_{S'}\boldsymbol{\epsilon} - \boldsymbol{\nu}^T(\mathbf{B} + \alpha_v \mathbf{M})\boldsymbol{\nu} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}(\delta_{ac} - \eta_o) \\ &+ \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e \dot{\boldsymbol{\nu}}\end{aligned} \tag{6.134}$$

However, such a choice will require exact values of the Jacobian matrix $\mathbf{J}_e$ and the full image state $\boldsymbol{\epsilon}$, which are not exactly known. Due to uncertainties on these variables, a robust design should be preferable.
Thus, let $\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)$ be defined as the upper bound of $(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)^T \mathbf{J}_e^T B_{S'}^T P_{S'}\boldsymbol{\epsilon}$ such that

$$\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t) \geq \left\|(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)^T\right\| \left\|\mathbf{J}_e^T\right\| \left\|B_{S'}^T P_{S'}\right\| \|\boldsymbol{\epsilon}\| \tag{6.135}$$

By defining the approximate Jacobian as $\widehat{\mathbf{J}}_e \triangleq \mathbf{J}_e - \widetilde{\mathbf{J}}_e$, where $\widetilde{\mathbf{J}}_e$ denotes the estimate error on the Jacobian. We can write the norm of $\widehat{\mathbf{J}}_e$ as follows

$$\left\|\widehat{\mathbf{J}}_e\right\| = \|\mathbf{J}_e\| \left\|\mathbf{I} - \mathbf{J}_e^{\dagger}\widetilde{\mathbf{J}}_e\right\| \tag{6.136}$$

Using (6.136), it can be shown that $\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)$ can be written as function of $\widehat{\mathbf{J}}_e$ as follows

$$\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t) = \frac{1}{1 - \kappa_1} \left\|(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)^T\right\| \left\|\widehat{\mathbf{J}}_e^T\right\| \left\|B_{S'}^T P_{S'}\right\| \|\boldsymbol{\epsilon}\| \tag{6.137}$$

with κ_1 a positive number less than 1 [126, p. 263]. Similarly to variable structure control, the proposed robust control input $\boldsymbol{\mu}$ is now design as follows

$$\boldsymbol{\mu} = \begin{cases} -\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)\frac{\boldsymbol{\nu}}{\|\boldsymbol{\nu}\|} & ; if \ \|\boldsymbol{\nu}\| > \varepsilon_1 \\ -\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)\frac{\boldsymbol{\nu}}{\varepsilon_1} & ; if \ \|\boldsymbol{\nu}\| \leq \varepsilon_1 \end{cases} \tag{6.138}$$

where $\varepsilon_1 > 0$ is used as boundary layer in order to avoid a chattering phenomenon [187]. The residual error can be made arbitrarily small by choosing smaller value of ε_1. It was shown in

[126, p. 264] that this kind of design, ensures that

$$\begin{aligned}\boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\boldsymbol{\nu} + \boldsymbol{\nu}^T\boldsymbol{\mu} \leq\ 0, \qquad \forall \|\boldsymbol{\nu}\| > \varepsilon_1 \\ \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e(\Lambda_v + \widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{J}}}_e)\boldsymbol{\nu} + \boldsymbol{\nu}^T\boldsymbol{\mu} \leq \frac{\rho(\boldsymbol{J}_e, \boldsymbol{\epsilon}, t)\varepsilon}{2}, \ \forall \|\boldsymbol{\nu}\| \leq \varepsilon_1\end{aligned} \tag{6.139}$$

Hence, based on (6.139), the following inequality holds

$$\begin{aligned}\dot{\mathcal{L}} \leq -\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}} Q_{S'}\boldsymbol{\epsilon} - \boldsymbol{\nu}^T(\mathbf{B} + \alpha_v\mathbf{M})\boldsymbol{\nu} + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}(\delta_{ac} - \eta_o) \\ + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{J}_e\dot{\boldsymbol{\nu}} + \frac{\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)\varepsilon_1}{2}\end{aligned} \tag{6.140}$$

Step 5. Design of the Feed-forward input ${}^c\widehat{V}_{o_{ff}}$

Consider now the third and fourth terms of (6.140), which can be lumped in one expression. Let us call it Υ. Hence, we have

$$\Upsilon = \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}(\delta_{ac} - \eta_o + \mathbf{J}_e\dot{\boldsymbol{\nu}}) \tag{6.141}$$

Recalling from Equation (6.125), that $\delta_{ac} = \Lambda_v \mathbf{L}_e {}^c\widehat{V}_{o_{ff}}$ and η_o can be rewritten as

$$\eta_o = \mathbf{L}_e({}^c\dot{V}_0 + \Lambda_v {}^cV_0) + \mathbf{J}_e\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e {}^cV_0 \tag{6.142}$$

At this stage, we can choose to split ${}^c\widehat{V}_{o_{ff}}$ into two components ${}^c\widehat{V}_{o_{ff1}}$ and ${}^c\widehat{V}_{o_{ff2}}$ such that one will compensate in a feed-forward manner the expression $({}^c\dot{V}_0 + \Lambda_v {}^cV_0)$ in (6.142), which has less uncertainties as compared to the expression $\mathbf{J}_e(\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e {}^cV_0 + \dot{\boldsymbol{\nu}})$, while the second components will compensate in a robust manner the remaining part of (6.141). This has the benefit to reduce the magnitude of saturated control actions due to the robust variable structure type controller, allowing as such smoother responses. Thus, Equation (6.141) can now be written as

$$\begin{aligned}\Upsilon = \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'} \mathbf{L}_e[\Lambda_v {}^c\widehat{V}_{o_{ff1}} - ({}^c\dot{V}_0 + \Lambda_v {}^cV_0)] \\ + \boldsymbol{\epsilon}^{\boldsymbol{T}} P_{S'} B_{S'}[\delta_{ac2} + \mathbf{J}_e(\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e {}^cV_0 + \dot{\boldsymbol{\nu}})]\end{aligned} \tag{6.143}$$

where $\delta_{ac2} = \mathbf{L}_e\Lambda_v {}^c\widehat{V}_{o_{ff2}}$. We can now design ${}^c\widehat{V}_{o_{ff1}}$ as follows

$${}^c\widehat{V}_{o_{ff1}} = \Lambda_v^{-1} {}^c\dot{\widehat{V}}_0 + {}^c\widehat{V}_0 \tag{6.144}$$

With this choice, it clear that the residual error of the first row of (6.144) will be negligible if we have a good estimate ${}^c\widehat{V}_0$ of the object's velocity.
Similarly to the control variable $\boldsymbol{\mu}$, to design δ_{ac2} we can start by defining an upper bound of the expression $\mathbf{J}_e(\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e {}^cV_0 + \dot{\boldsymbol{\nu}})$ in the second row of (6.143), we have

$$\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t) \geq \|\mathbf{J}_e\| \left\|(\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e {}^cV_0 + \dot{\boldsymbol{\nu}})\right\| \tag{6.145}$$

Using (6.136), $\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t)$ can be determined as

$$\begin{aligned}\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t) &= \frac{1}{1-\kappa_2}\left\|\widehat{\mathbf{J}}_e\right\|\left\|(\widehat{\mathbf{J}}_e^{\dagger}\dot{\widehat{\mathbf{L}}}_e{}^cV_0 + \dot{\boldsymbol{\nu}})\right\| \\ &= \frac{1}{1-\kappa_2}\left[\left\|\dot{\widehat{\mathbf{L}}}_e\right\|\|{}^cV_0\| + \left\|\widehat{\mathbf{J}}_e\right\|\|\dot{\boldsymbol{\nu}}\|\right]\end{aligned} \tag{6.146}$$

where κ_2 is defined as κ_1 above. The robust control input δ_{ac2} is now designed as follows

$$\delta_{ac2} = \begin{cases} -\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t)\frac{B_{S'}^T P_{S'}\boldsymbol{\epsilon}}{\|B_{S'}^T P_{S'}\boldsymbol{\epsilon}\|} & ; if \ \left\|B_{S'}^T P_{S'}\boldsymbol{\epsilon}\right\| > \varepsilon_2 \\ -\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t)\frac{B_{S'}^T P_{S'}\boldsymbol{\epsilon}}{\varepsilon_2} & ; if \ \left\|B_{S'}^T P_{S'}\boldsymbol{\epsilon}\right\| \leq \varepsilon_2 \end{cases} \tag{6.147}$$

where ε_2 has the same definition as ε_1. Hence, ${}^c\widehat{V}_{o_{ff2}}$ can be determined as

$${}^c\widehat{V}_{o_{ff2}} = \mathbf{L}_e^{\dagger}\Lambda_v^{-1}\delta_{ac2} \tag{6.148}$$

From (6.144) and (6.148), ${}^c\widehat{V}_{o_{ff}} = {}^c\widehat{V}_{o_{ff1}} + {}^c\widehat{V}_{o_{ff2}}$ will be given by

$${}^c\widehat{V}_{o_{ff}} = \Lambda_v^{-1}({}^c\dot{\widehat{V}}_0 + \Lambda_v{}^c\widehat{V}_0) + \Lambda_v^{-1}\mathbf{L}_e^{\dagger}\delta_{ac2} \tag{6.149}$$

Finally, according to (6.149), the feed-forward input as applied $\widehat{\mathbf{L}}_e{}^c\widehat{V}_{o_{ff}}$ will be given by

$$\widehat{\mathbf{L}}_e{}^c\widehat{V}_{o_{ff}} = \Lambda_v^{-1}\left[\widehat{\mathbf{L}}_e({}^c\dot{\widehat{V}}_0 + \Lambda_v{}^c\widehat{V}_0) + \delta_{ac2}\right] \tag{6.150}$$

Note that the product $\widehat{\mathbf{L}}_e\mathbf{L}_e^{\dagger}$ in (6.150) has been replaced by its ideal value $\mathbf{I}$, in order to obtain a well-defined control input.

Using control law (6.150), it can be shown that the time-derivative of the Lyapunov function $\dot{\mathcal{L}}$ will now satisfy the following inequality

$$\begin{aligned}\dot{\mathcal{L}} \leq -\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}}Q_{S'}\boldsymbol{\epsilon} - \boldsymbol{\nu}^T(\mathbf{B} + \alpha_v\mathbf{M})\boldsymbol{\nu} + \frac{\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)\varepsilon_1}{2} \\ + \frac{\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t)\varepsilon_2}{2}\end{aligned} \tag{6.151}$$

The negative-definiteness of $\dot{\mathcal{L}}$ ($\dot{\mathcal{L}} < 0$) will be guaranteed provided that

$$\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}}Q_{S'}\boldsymbol{\epsilon} + \boldsymbol{\nu}^T(\mathbf{B} + \alpha_v\mathbf{M})\boldsymbol{\nu} > \frac{\rho(\mathbf{J}_e, \boldsymbol{\epsilon}, t)\varepsilon_1}{2} + \frac{\varrho(\mathbf{J}_e, {}^cV_0, \dot{\boldsymbol{\nu}}, t)\varepsilon_2}{2} \tag{6.152}$$

Dropping the variables of ρ and ϱ, (6.152) can be rewritten as

$$X_{\epsilon\nu}^T\boldsymbol{Q}_{QBM_\alpha}X_{\epsilon\nu} > \frac{\rho\varepsilon_1 + \varrho\varepsilon_2}{2} \tag{6.153}$$

where

$$X_{\epsilon\nu} = \begin{bmatrix} \boldsymbol{\epsilon} \\ \boldsymbol{\nu} \end{bmatrix} \text{ and } \boldsymbol{Q}_{QBM} = \begin{bmatrix} \frac{Q_{S'}}{2} & 0 \\ 0 & (\mathbf{B} + \alpha_v \mathbf{M}) \end{bmatrix}$$

Using the Euclidean norm, Equation (6.153) will satisfy the following relation

$$\lambda_{min}(\boldsymbol{Q}_{QBM_\alpha}) \left\| X_{\epsilon\nu} \right\|^2 \geq \frac{\rho\varepsilon_1 + \varrho\varepsilon_2}{2} \tag{6.154}$$

which can be rewritten as

$$\left\| X_{\epsilon\nu} \right\| \geq \left(\frac{\rho\varepsilon_1 + \varrho\varepsilon_2}{2\lambda_{min}(\boldsymbol{Q}_{QBM_\alpha})} \right)^{\frac{1}{2}} \tag{6.155}$$

Equation (6.155) shows the uniform ultimate boundedness [126, p. 264] of all solutions of the closed loop visual servoing system.

6.7 Conclusion

In this chapter, after reviewing and discussing different approaches related to dynamic visual servoing, we have proposed an approach consisting first of improving the reference trajectories generated by the visual controller and then improving the tracking performances of the robot which executes these commands.

Regarding the visual controller, from the classical law achieving an exponential decrease of the task function, we have derived two new laws by using a simple low-pass filter either on features error before computing the command velocity or on the velocity itself. This has the benefit of yielding faster and, damped second order response. Then, we have derived the model of the resulting closed loop. Using perturbation theory, we have carried out the stability and convergence analysis of the system. This analysis has also been extended to the case of moving target. Furthermore, the dynamic interaction between the vision system and the robot has been considered.

Finally, using Lyapunov's second method, we have designed a robust adaptive controller and proved the ultimate uniform boundedness of the closed loop solution. In the next chapter, we provide simulation as well as experimental results validating our approach.

Chapter 7

Experimental Validation of Proposed Dynamic Visual Servoing Laws

This chapter presents simulation and experimental results to validate the dynamic visual servoing framework developed in the previous chapter. First of all, the proposed control laws (6.40) and (6.86) are simulated and compared to classical visual servoing control laws. Afterward, extensive simulations are carried out to empirically estimate the range of their parameters giving the best performances. Finally, an experimental validation is carried out on humanoid robot NAO.

7.1 Simulations Results

The simulation results presented here have been obtained with a visual servoing algorithm we wrote in Matlab and included in the CD accompanying this thesis. The camera has six DOFs and an eye-in-hand configuration. Once again, a square (side $L = 0.20\,m$) formed by four coplanar points is chosen as the target object.

In our formulation, since no particular assumptions were made regarding the interaction matrix, a simple IBVS scheme will be used here with the points coordinates as features. Thus, the interaction matrix used has the following form

$$L_s = \begin{bmatrix} -\frac{1}{Z_1} & 0 & \frac{x_1}{Z_1} & x_1y_1 & -(1+x_1^2) & y_1 \\ -\frac{1}{Z_2} & 0 & \frac{x_2}{Z_2} & x_2y_2 & -(1+x_2^2) & y_2 \\ -\frac{1}{Z_3} & 0 & \frac{x_3}{Z_3} & x_3y_3 & -(1+x_3^2) & y_3 \\ -\frac{1}{Z_4} & 0 & \frac{x_4}{Z_4} & x_4y_4 & -(1+x_4^2) & y_4 \\ 0 & -\frac{1}{Z_1} & \frac{y_1}{Z_1} & 1+y_1^2 & -x_1y_1 & -x_1 \\ 0 & -\frac{1}{Z_2} & \frac{y_2}{Z_2} & 1+y_2^2 & -x_2y_2 & -x_2 \\ 0 & -\frac{1}{Z_3} & \frac{y_3}{Z_3} & 1+y_3^2 & -x_3y_3 & -x_3 \\ 0 & -\frac{1}{Z_4} & \frac{y_4}{Z_4} & 1+y_4^2 & -x_4y_4 & -x_4 \end{bmatrix} \tag{7.1}$$

To simplify notation, we will use FE (Filtered Error) and FV (Filtered Velocity) to refer re-

spectively to control law (6.40) and (6.86). These laws were derived particularly to speed up the servoing and provide reference velocity and acceleration for the lower-level robot's controller. In that respect, the settling time and the maximum velocity required to achieve a task will be considered as evaluation and comparison variables.

7.1.1 Comparison Between Classical, Filtered Error Based and Filtered Velocity Based Visual Servoing Laws

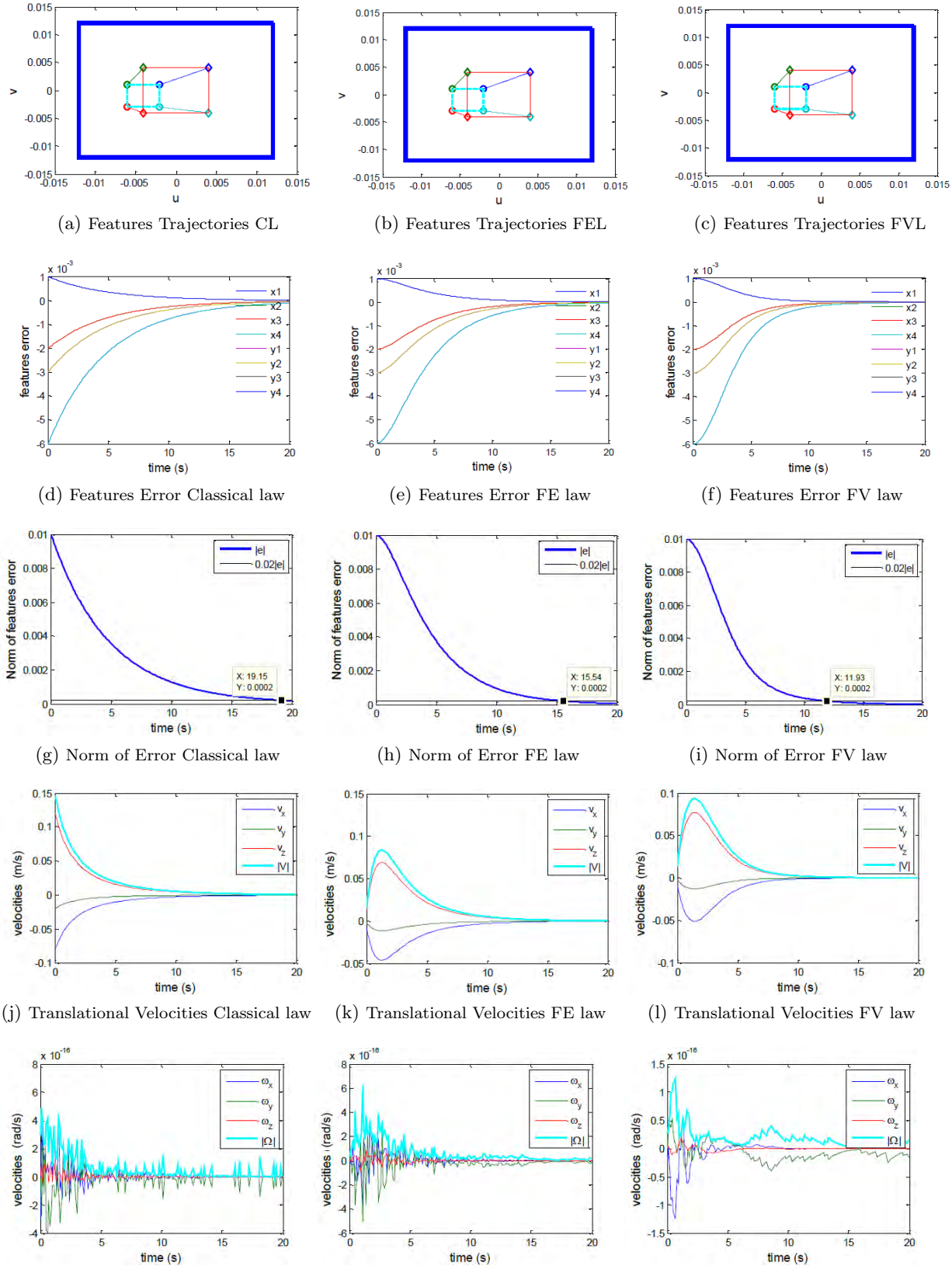
For comparison purposes, we will consider three cases requiring from the camera, respectively, a simple translation, a simple rotation and a general motion. Also, to compare these laws in their respective best conditions, the subsequent simulations will assume knowledge of the exact depths of feature points.

7.1.1.1 Case 1: Translational Motion of the Camera

Consider a servoing task where the camera, in the world frame, is initially located at $(-0.50,\ 0.2,\ 0.35)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$ and the desired pose of the camera relative to the object is given by $(-0.30,\ 0.0, 0.0)\,m$ and $(\frac{\pi}{2},\ 0.0,\ \frac{\pi}{2})\,rad$. The object is fixed and located at $(0.10,\ 0.0,\ 0.30)\,m$ and $(0.0,\ 0.0,\ 0.0)\,rad$ with respect to the world reference frame. The interaction matrix is computed at each iterations. The gains are set as follows: $K_p = 0.4$ and $K_v = 0.8$.

The results related to this simulation are grouped and shown in Figure 7.1, where the first, the second, and the third column correspond respectively to the classical law, the FE law, and the FV law. In the first row are shown the trajectories of the features points as they move from their initial positions, represented by small circles and a cyan frame, to their desired positions indicated by small diamonds and a red frame. In the second row, the feature points error, representing the task function is shown. Its Euclidean norm and a line indicating the $2\,\%$ of its value are plotted in the following row. In the fourth and the fifth row are shown, respectively, the translational velocities and the rotation velocities with their norms.

An improvement in settling time for the FE and FV laws can be noticed since the norm of the task function crosses the $2\,\%$ line at about $19.15\,s$ for the classical law, $15.54\,s$ for FE and $11.93\,s$ for FV. Thus, as expected, the classical law has the longest settling time, yet its velocities reached the highest values.
In the image space, the straight line trajectories of the classical law, particularly due to the choice of the exact interaction matrix, are preserved for the FE law. However, for FV some curvatures are perceptible. They could be explained by the fact that the interaction matrix in this scheme is directly filtered, altering as such its value from the exact one.



(a) Features Trajectories CL (b) Features Trajectories FEL (c) Features Trajectories FVL

(d) Features Error Classical law (e) Features Error FE law (f) Features Error FV law

(g) Norm of Error Classical law (h) Norm of Error FE law (i) Norm of Error FV law

(j) Translational Velocities Classical law (k) Translational Velocities FE law (l) Translational Velocities FV law

(m) Rotational Velocities Classical law (n) Rotational Velocities FE law (o) Rotational Velocities FV law

Figure 7.1: Comparison for translational motion of the camera

7.1.1.2 Case 2: Rotational Motion of the Camera

Consider now another task, where the initial and the desired pose of the camera are only different in orientation. The pose of the target stays the same as previously but the camera pose is now $(-0.30,\, 0.0,\, 0.30)\, m$ and $(\frac{3\pi}{5},\, -\frac{\pi}{3},\, \frac{3\pi}{10})\, rad$ and its desired pose relative to the target is given by $(-0.40,\, 0.0,\, 0.00)\, m$ and $(\frac{\pi}{2},\, 0.0,\, \frac{\pi}{2})\, rad$. The results of this simulation are shown in Figure 7.2. Although the settling time of the FV law has shifted towards that of FE, it can be seen that these laws exhibited faster responses with smaller maximum velocities than the classical law.

7.1.1.3 Case 3: General Motion of the Camera

Finally, consider a task requiring a general displacement of the camera in order to be performed. The target configuration is given by $(0.10,\, 0.0,\, 0.30)\, m$ and $(\frac{\pi}{3},\, \frac{\pi}{3},\, \frac{\pi}{5})\, rad$, the camera configuration is given by $(-0.50,\, 0.3,\, 0.55)\, m$ and $(\frac{\pi}{2},\, 0.0,\, \frac{\pi}{2})\, rad$ and its desired pose with respect the target is $(-0.40,\, 0.0,\, 0.00)\, m$ and $(\frac{\pi}{2},\, 0.0,\, \frac{\pi}{2})\, rad$.

In view of the results of this simulation, which are grouped in Figure 7.3, the comments made for previous simulations are also valid here.

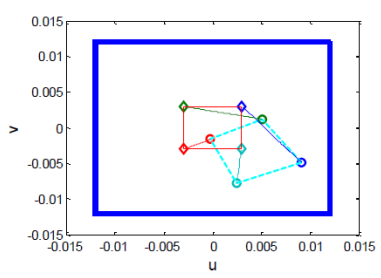
It is important to note the consistency of settling times with the classical and the FE laws. The FV law on the other hand has the shortest settling time in simple translation but this time increases towards that of FE as the camera's rotation becomes more involved.

7.1.2 Effects of K_v in Servoing Loop

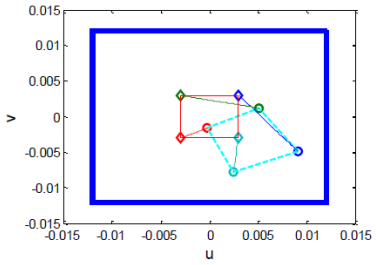
The previous simulations were conducted with a fixed K_v (0.8) and the exact interaction matrix L_s. Now the performances of the servoing loop for different values of K_v while considering approximated interaction matrices will be analyzed. Hence, we will conduct the three following experiments:

- in the first, K_v will be varied while considering the exact interaction matrix $L_s = L_s(t)$.
- in the second, the exact interaction matrix is replaced by its average with the interaction matrix at the desired pose, $L_s = \frac{1}{2}\left(L_s(t) + L_{s^*}\right)$.
- in the third, the interaction matrix is coarsely approximated with its fixed value at the desired pose, $L_s = L_{s^*}$.

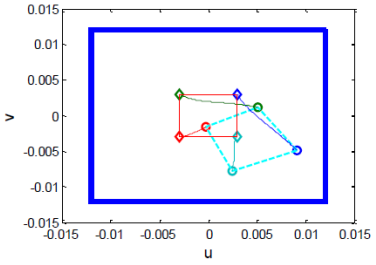
Given that the decrease of the task function with a classical law is consistent and depends only on K_p, we will define the variation of K_v relatively to K_p. To that end, let α_{pv} be defined as the ratio between K_p and K_v, $\alpha_{pv} = \frac{K_v}{K_p}$, such that $\alpha_{pv} = \infty$ corresponds to the classical law. Hence, in the simulations, the value of α_{pv} will vary from ∞ to 0.5.



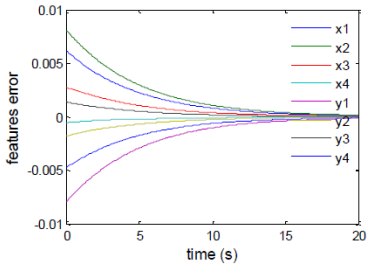
(a) Features Trajectories CL



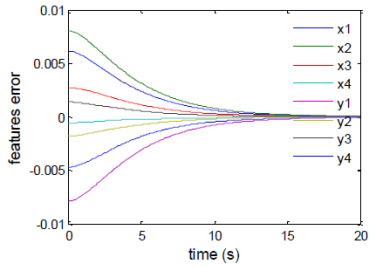
(b) Features Trajectories FEL



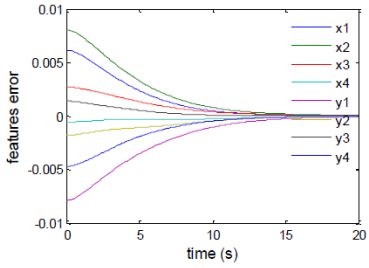
(c) Features Trajectories FVL



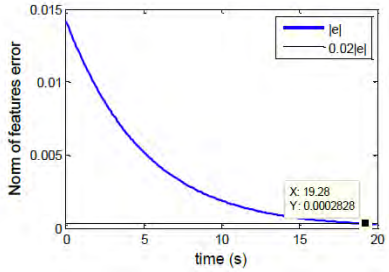
(d) Features Error Classical law



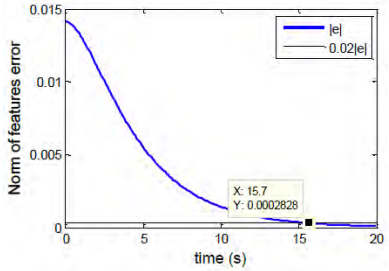
(e) Features Error FE law



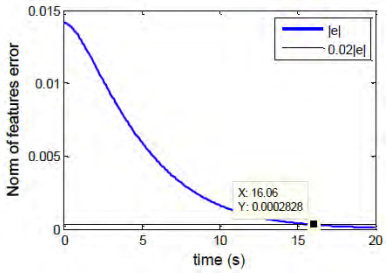
(f) Features Error FV law



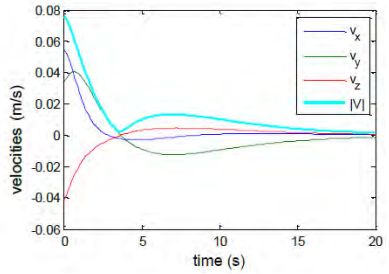
(g) Norm of Error Classical law



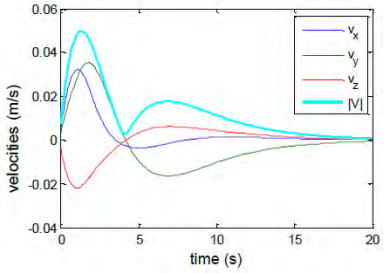
(h) Norm of Error FE law



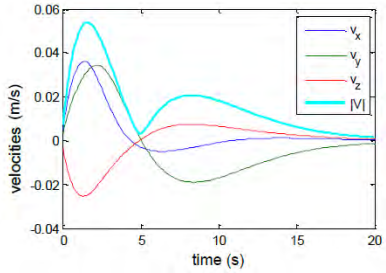
(i) Norm of Error FV law



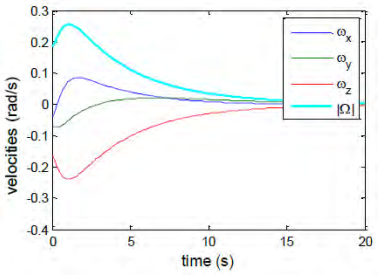
(j) Translational Velocities Classical law



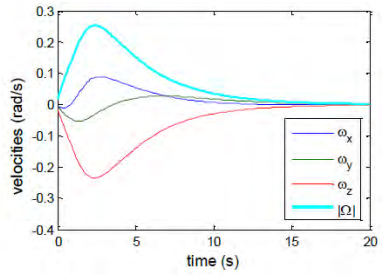
(k) Translational Velocities FE law



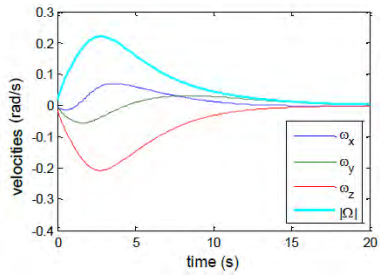
(l) Translational Velocities FV law



(m) Rotational Velocities Classical law

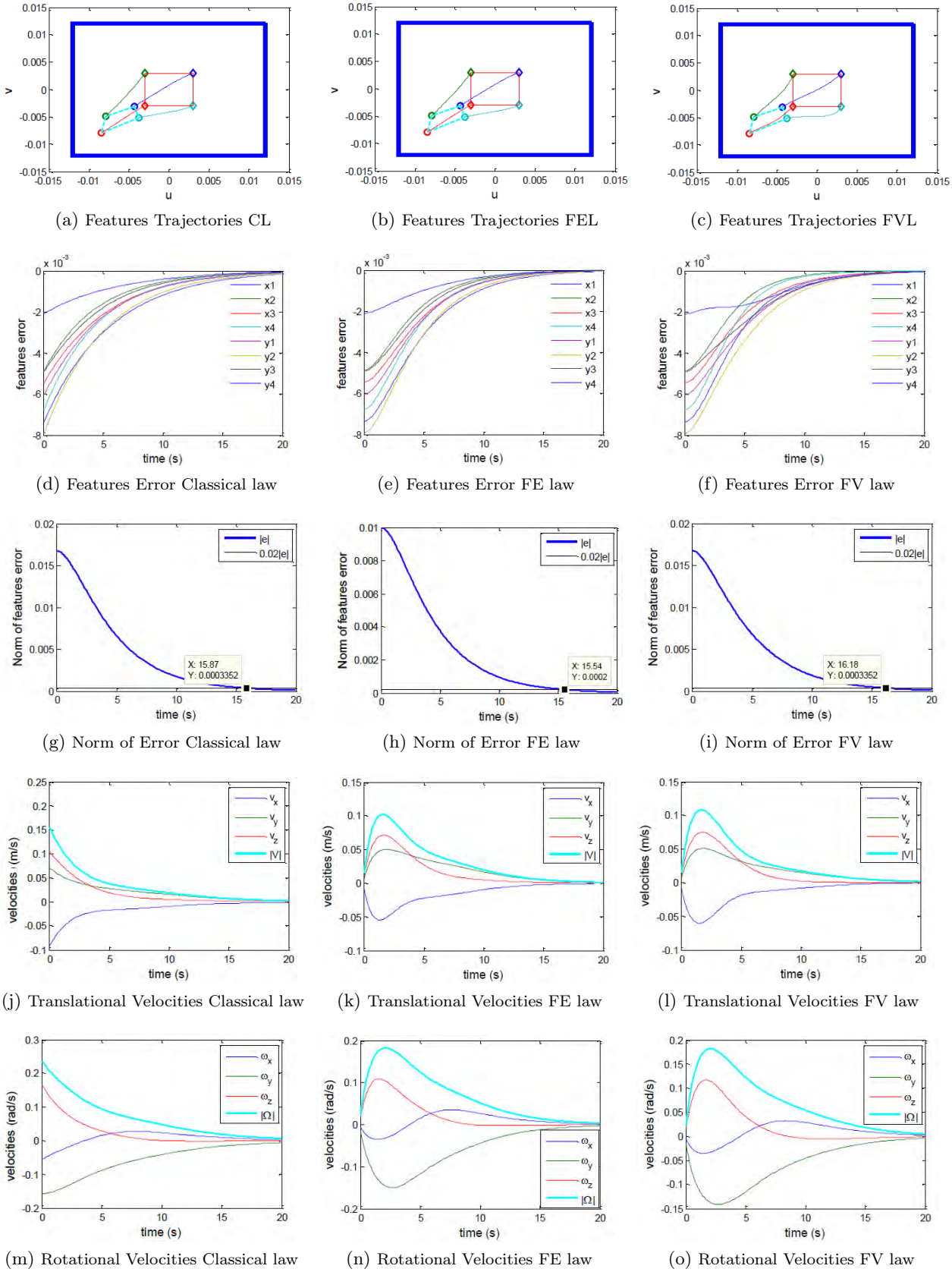


(n) Rotational Velocities FE law



(o) Rotational Velocities FV law

Figure 7.2: Comparison for rotational motion of the camera



(a) Features Trajectories CL (b) Features Trajectories FEL (c) Features Trajectories FVL

(d) Features Error Classical law (e) Features Error FE law (f) Features Error FV law

(g) Norm of Error Classical law (h) Norm of Error FE law (i) Norm of Error FV law

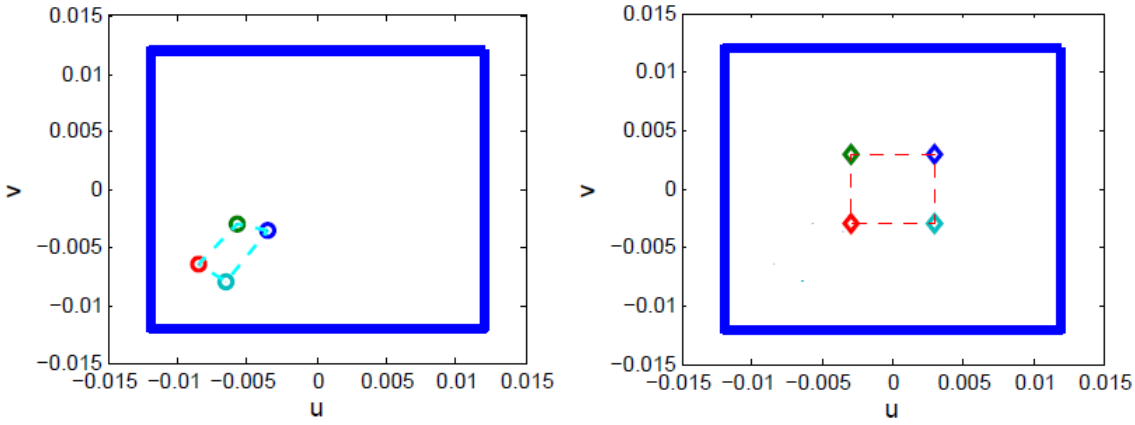
(j) Translational Velocities Classical law (k) Translational Velocities FE law (l) Translational Velocities FV law

(m) Rotational Velocities Classical law (n) Rotational Velocities FE law (o) Rotational Velocities FV law

Figure 7.3: Comparison for general motion of the camera

7.1.2.1 Task Specification

To carry out this analysis, let us also consider a positioning task requiring a general displacement of the camera. The target's configuration is $(0.10,\, 0.0,\, 0.30)\, m$ and $(0.0,\, \frac{\pi}{4},\, \frac{\pi}{5})\, rad$, the initial camera pose is $(0.10,\, 0.0,\, 0.30)\, m$ and $(\frac{\pi}{2},\, 0.0,\, \frac{\pi}{2})\, rad$, and its desired pose relative to the camera is given by $(0.10,\, 0.0,\, 0.30)\, m$ and $(\frac{\pi}{2},\, 0.,\, \frac{\pi}{2})\, rad$.
The initial and desired image configurations are illustrated in Figure 7.4.



(a) Initial image features

(b) Desired Image features

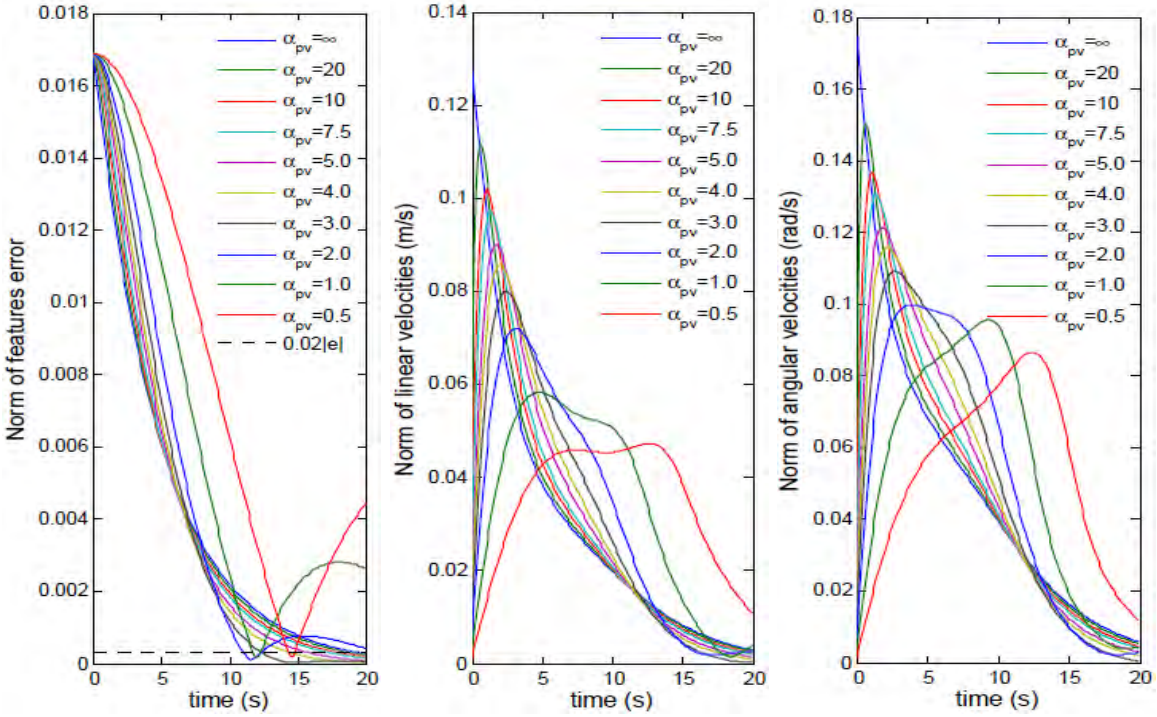
Figure 7.4: Initial and desired image features configuration

7.1.2.2 Case 1: Exact interaction matrix $L_s = L_s(t)$

Focusing only on the norms of task function and the norms of translational and rotational velocities, the results of this simulation are shown in Figure 7.5 and Figure 7.6 for the FE and FV laws, respectively. The important values on their side are recorded in Table 7.1.

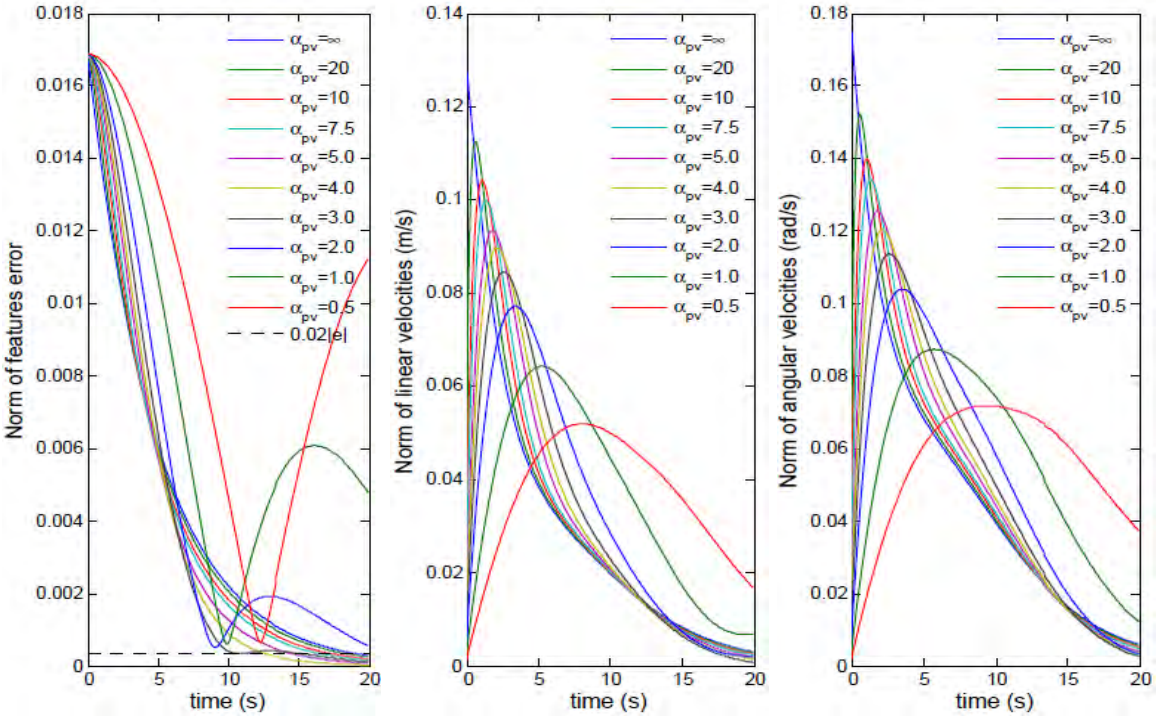
It can be seen that the fastest response without overshoot is obtained for $\alpha_{pv} \approx 3.0$. With this value, the settling time of the classical control law can be shortened by 45 %.

For all values of $\alpha_{pv} > 3.0$, the task function settles without oscillations. The oscillations can be noticed by the change of direction of the task function norm, for instance the bouncing back of the green or red plot in Figure 7.5a corresponds to overshoots of some features. As α_{pv} decreases below 3.0 and lower, the response becomes slower and more oscillatory.



(a) Norm of Task Function (b) Norm of Trans Velo (c) Norm of Rot Velo

Figure 7.5: Variation of α_{pv} with exact interaction matrix and the FE law



(a) Norm of Task Function (b) Norm of Trans Velo (c) Norm of Rot Velo

Figure 7.6: Variation of α_{pv} with exact interaction matrix and the FV law

α_{pv}	Settling time 2% $[s]$		max |V| $[m/s]$		max |Ω| $[rad/s]$	
	FE	FV	FE	FV	FE	FV
∞	19.25	19.25	0.1274	0.1274	0.1745	0.1745
20	18.65	18.47	0.1115	0.1126	0.1503	0.1520
10	17.74	17.41	0.1019	0.1041	0.1368	0.1399
7.5	17.14	16.61	0.0972	0.0999	0.1306	0.1341
5.0	15.64	14.64	0.0900	0.0935	0.1213	0.1254
4.0	14.34	12.61	0.0857	0.0897	0.1160	0.1204
3.0	12.21	10.59	0.0801	0.0846	0.1091	0.1137
2.0	>20 *	>20 *	0.0720	0.0772	0.0998	0.1040
1.0	>20 *	>20 *	0.0583	0.0643	0.0956	0.0874
0.5	>20 *	>20 *	0.0474	0.0519	0.0865	0.0718

Table 7.1: Summary of results for variation of $\alpha_{pv}$with exact L_s for the FE and FV laws

7.1.2.3 Case 2: Average interaction matrix $L_s = \frac{1}{2}\left(L_s(t) + L_{s^*}\right)$

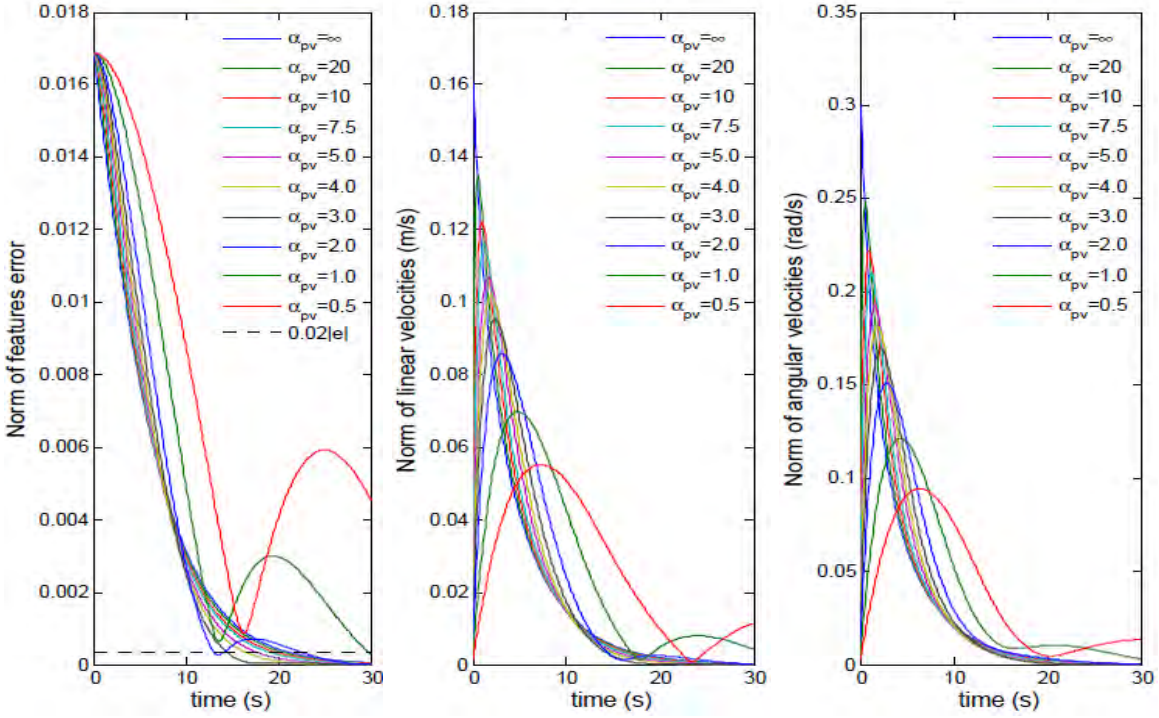
In the stability and convergence analysis (Section 6.4.3) of the proposed laws, it was shown that a choice of $\Lambda_v \gg \left((L_e\dot{\widehat{L}}_e^{\dagger})(L_e\widehat{L}_e^{\dagger})^{+}\right)$ for the FE law will guarantee the convergence of the task. The more accurate L_s is, the faster will be the convergence of the closed loop.

As shown in Figures 7.7 and 7.8, and summarized in Table 7.2, this experiment illustrates how the convergence of the system, with inaccurate L_s, is affected by different values of α_{pv}.

When compared to case 1, one can see that the system is a bit slower and requires higher velocities for similar values of α_{pv}. The best results are still obtained for $\alpha_{pv} \approx 3.0$. Also, one can notice that, for $\alpha_{pv} < 3.0$, the FV law task function presents smaller oscillations.

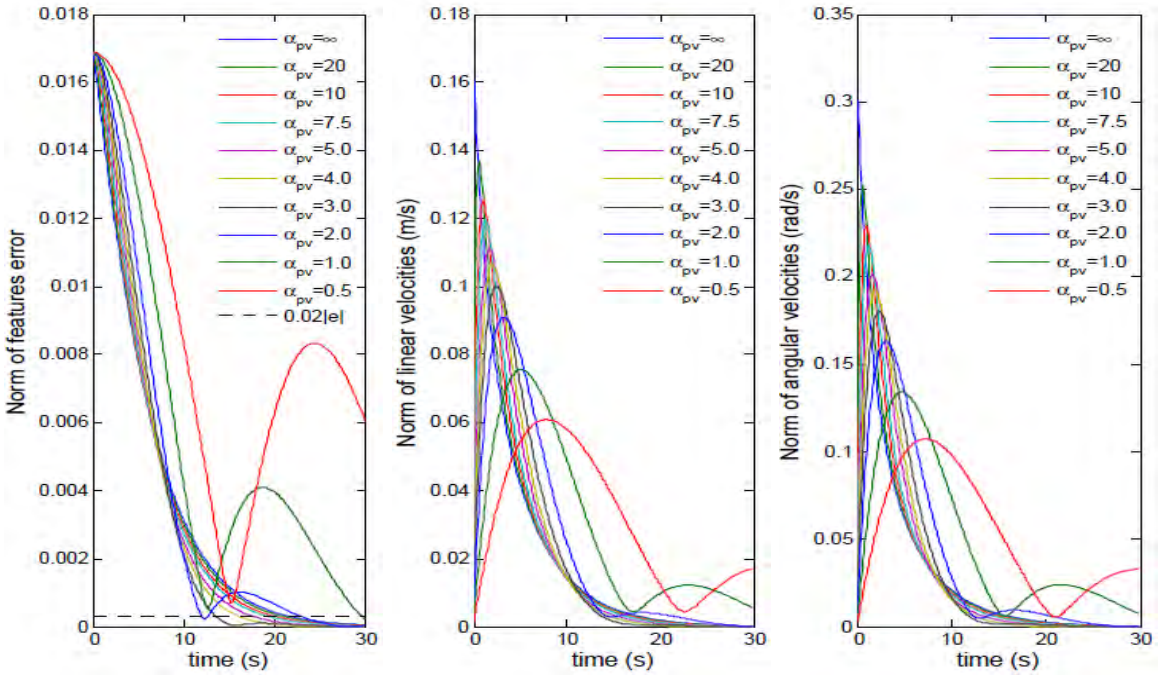
α_{pv}	Settling time 2% $[s]$		max |V| $[m/s]$		max |Ω| $[rad/s]$	
	FE	FV	FE	FV	FE	FV
∞	21.20	21.20	0.1605	0.1605	0.3002	0.3002
20	20.62	20.55	0.1350	0.1370	0.2483	0.2522
10	19.75	19.63	0.1220	0.1251	0.2220	0.2284
7.5	19.10	18.95	0.1160	0.1195	0.2099	0.2178
5.0	17.74	18.84	0.1070	0.1112	0.1923	0.2016
4.0	16.44	17.34	0.1019	0.1065	0.1824	0.1923
3.0	14.44	13.43	0.0952	0.1001	0.1694	0.1802
2.0	22.56	22.56	0.0857	0.0910	0.1512	0.1630
1.0	>30 *	>30 *	0.0699	0.0756	0.1213	0.1342
0.5	>30 *	>30 *	0.0552	0.0609	0.0945	0.1074

Table 7.2: Summary of results for variation of $\alpha_{pv}$with average L_s for the FE and FV laws



(a) Norm of Task Function (b) Norm xxof Trans Velo (c) Norm of Rot Velo

Figure 7.7: Variation of α_{pv} with average interaction matrix and the FE law



(a) Norm of Task Function (b) Norm of Trans Velo (c) Norm of Rot Velo

Figure 7.8: Variation of α_{pv} with average interaction matrix and the FV law

7.1.2.4 Case 3: Interaction matrix at desired pose $L_s = L_{s^*}$

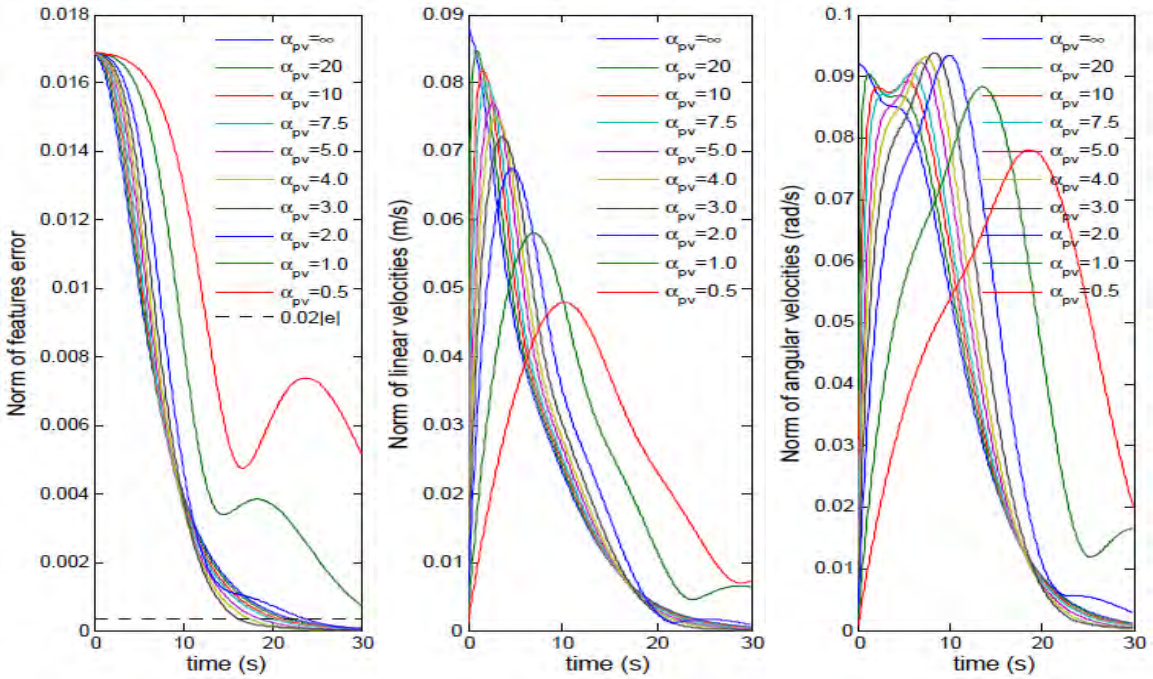
In this simulation, the inaccuracy on L_s is reinforced by choosing a local approximation of the actual interaction matrix. Similarly to the previous cases, the results are shown and synthesized respectively in Figures 7.9 and 7.10, and in Table 7.3.

As expected, the results of FE and FV are identical since L_s is constant. The FE and FV laws are still settling faster than the Classical law. However, this improvement is limited in an approximate range of 25 %.
When compared to the two previous cases, the system is a bit slower, but it requires the lowest maximum velocities. Once again, the best results are obtained for $\alpha_{pv} \approx 3.0$. Smaller values than the latter yield more oscillating responses.

α_{pv}	Settling time 2% $[s]$		max |V| $[m/s]$		max |Ω| $[rad/s]$	
	FE	FV	FE	FV	FE	FV
∞	22.36	22.36	0.0881	0.0881	0.0919	0.0919
20	21.65	21.65	0.0848	0.0848	0.0903	0.0903
10	20.85	20.85	0.0819	0.0819	0.0892	0.0892
7.5	20.15	20.15	0.0801	0.0801	0.0905	0.0905
5.0	18.75	18.75	0.0771	0.0771	0.0923	0.0923
4.0	17.54	17.54	0.0751	0.0751	0.0931	0.0931
3.0	16.24	16.24	0.0722	0.0722	0.0938	0.0938
2.0	23.76	23.76	0.0674	0.0674	0.0934	0.0934
1.0	>30 *	>30 *	0.0581	0.0581	0.0883	0.0883
0.5	>30 *	>30 *	0.0480	0.0480	0.0781	0.0781

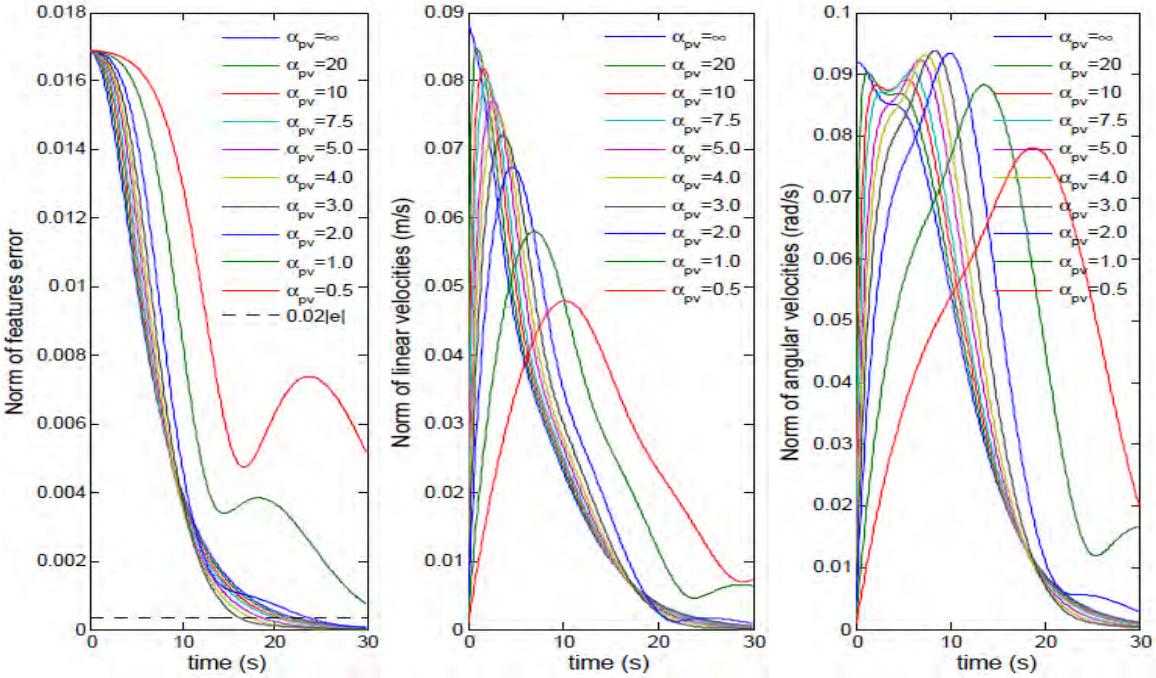
Table 7.3: Summary of results for variation of $\alpha_{pv}$with desired L_s for the FE and FV laws

Note. As summary of the above simulations, we can affirm that by varying α_{pv}, the closed loop response can vary from first order response ($\alpha_{pv} = \infty$), to critically damped second order ($\alpha_{pv} \approx 3.0$) or even under-damped second order response ($\alpha_{pv} < 3.0$). This could be explained by the interaction between the poles introduced by the low-pass filter and that of the servoing system. The settling time can be sped up by 47 %. The better the approximate of L_s is, the faster the response will be.



(a) Norm of Task Function (b) Norm of Trans Velo (c) Norm of Rot Velo

Figure 7.9: Variation of α_{pv} with desired interaction matrix and the FE law



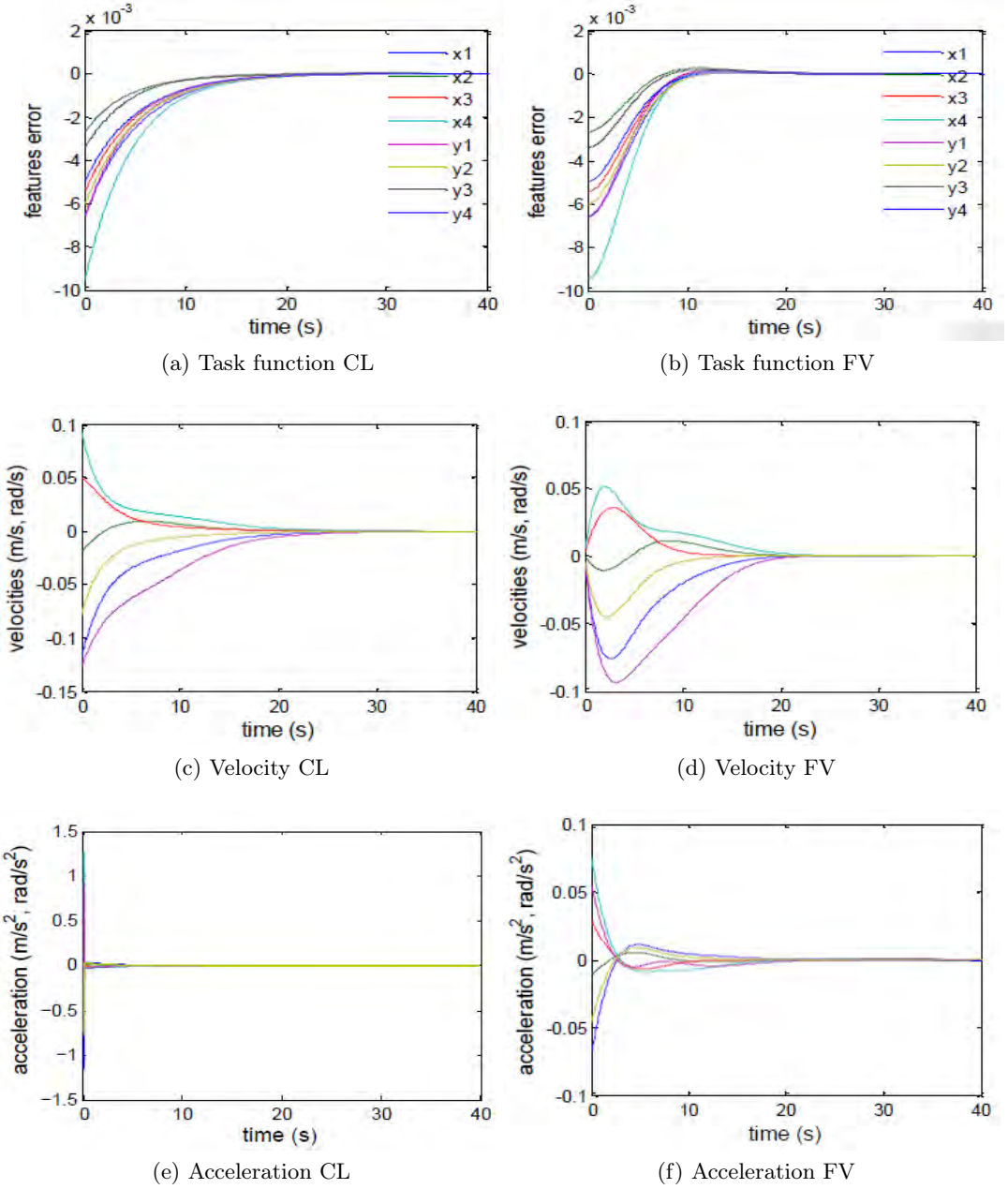
(a) Norm of Task Function (b) Norm of Trans Velo (c) Norm of Rot Velo

Figure 7.10: Variation of α_{pv} with average interaction matrix and the FV law

7.1.2.5 Reference Trajectories for the Robot Controller

Based on the results of the previous subsection, let us now plot the task function and its corresponding reference velocity and acceleration for the low-level robot controller. We consider the previous servoing task with the exact L_s and with $\Lambda_p = 0.4$ and $\alpha_{pv} = 3.0$.

As shown in Figure 7.11, comparing FV to the classical servoing law, we obtain smooth and faster response especially with smaller initial velocity and much smaller initial acceleration.



(a) Task function CL

(b) Task function FV

(c) Velocity CL

(d) Velocity FV

(e) Acceleration CL

(f) Acceleration FV

Figure 7.11: Reference trajectories for the classical and FV laws

7.2 Experimental Results

After validating in simulation the proposed visual servoing laws and showing how for a given task, the generated the control signals namely the reference acceleration and the reference velocity are smoother and lower than their equivalents with classical visual servoing laws, we will now validate them through actual experiments.

These experiments were carried out on NAO humanoid robot, whose hardware and software platforms have been presented previously (Section 5.1). NAO robot is only controlled in position and does not allow direct torque control. Hence, in the experiments, we will assume first that the position controller is well designed and can accurately track the references signals. Afterward, we will relax this assumption and apply the computed-torque based control scheme presented in Section 6.6.2 using the input transformation proposed in Section 6.3.

7.2.1 Dynamic Visual Servoing of the Humanoid's Head

Similarly to all previous experiments, visual positioning and tracking will be considered. Before performing them, let us derive first the head's dynamics.

7.2.1.1 Head's Dynamics

For simplicity, the head is modeled with respect to the torso frame and all dynamic interactions with the other body limbs are not considered. Following the development of Section 6.3, the model combining the humanoid head's link and actuators dynamics can be written as

$$[D(\mathbf{q}) + J]\,\ddot{\mathbf{q}} + [C(\mathbf{q},\dot{\mathbf{q}}) + B]\,\dot{\mathbf{q}} + G(\mathbf{q}) + F(\dot{\mathbf{q}}) + T_d = \boldsymbol{u} \tag{7.2}$$

where $\mathbf{q} = [\ q_1 \quad q_2\ ]^T$.

On the link side, the elements of $D(\mathbf{q})$ are given by

$$\begin{cases} D_{11} &= (m_h l_{hz}^2 + I_{hx})\sin(q_2))^2 + I_{hz}\cos(q_2)^2 - 2I_{hxz}\sin(q_2)\cos(q_2) \\ D_{12} &= I_{hyz}\cos(q_2) - I_{hxy}\sin(q_2) \\ D_{21} &= I_{hyz}\cos(q_2) - I_{hxy}\sin(q_2) \\ D_{22} &= m_h l_{hz}^2 + I_{hy} \end{cases}, \tag{7.3}$$

the $C(\mathbf{q},\dot{\mathbf{q}})$ matrix is given by

$$C(\mathbf{q},\dot{\mathbf{q}}) = \begin{bmatrix} C_{111}\dot{q}_1 + C_{211}\dot{q}_2 & C_{121}\dot{q}_1 + C_{221}\dot{q}_2 \\ C_{112}\dot{q}_1 + C_{212}\dot{q}_2 & C_{122}\dot{q}_1 + C_{222}\dot{q}_2 \end{bmatrix} \tag{7.4}$$

with C_{111}, C_{211}, C_{121}, C_{212}, C_{122}, and $C_{222} = 0$ and

$$\begin{cases} C_{112} & = \sin(q_2)\cos(q_2)(-m_h l_{hz}^2 - I_{hx} + I_{hz}) + I_{hxz}(2\cos(q_2)^2 - 1) \\ C_{221} & = -(I_{hyz}\sin(q_2) + I_{hxy}\cos(q_2)) \end{cases}. \tag{7.5}$$

The gravity vector $G(\mathbf{q})$ is given by

$$\begin{cases} G_1 & = 0 \\ G_2 & = -m_h l_{hz} g \sin(q_2) \end{cases}. \tag{7.6}$$

In the above equations, the nominal values of the parameters as provided by the manufacturer [204] are summarized in Table 7.4.

Label	***Parameters***	***Value***	***Unit***
m_h	mass of the head	0.66975	kg
l_{hz}	$x$position of the head CoM	0.05258	m
I_{hx}	head inertia about X axis	0.00263129518	$kg.m^2$
I_{hy}	head inertia about Y axis	0.00249112488	$kg.m^2$
I_{hz}	head inertia about Z axis	0.00099126938	$kg.m^2$
I_{hxy}	head product of inertia about XY axes	0.00000878814	$kg.m^2$
I_{hyz}	head product of inertia about YZ axes	0.00004098466	$kg.m^2$
I_{hxz}	head product of inertia about XZ axes	-0.00002995792	$kg.m^2$

Table 7.4: Head link's parameters

The friction torque $F(\dot{\mathbf{q}})$ can be modeled by the following dynamics

$$F(\dot{\mathbf{q}}) = F_v\dot{\mathbf{q}} + F_c tanh(\alpha\dot{\mathbf{q}}) \tag{7.7}$$

where F_v and F_c are the coefficients respectively of the viscous friction and Coulomb friction. $tanh$ and α are used instead of the function $signum$ in order to avoid discontinuity.

On the actuator side, the parameters J and B in (7.2) are given by

$$J = \begin{bmatrix} r_1^2 J_{a1} & 0 \\ 0 & r_2^2 J_{a2} \end{bmatrix} \text{ and } B = \begin{bmatrix} r_1^2\left(B_{a1} + \frac{K_{b1}K_{m1}}{R_1}\right) & 0 \\ 0 & r_2^2\left(B_{a2} + \frac{K_{b2}K_{m2}}{R_2}\right) \end{bmatrix} \tag{7.8}$$

where the gear ratios $r_1 = 150.27$ and $r_2 = 173,22$ [205]. The nominal values of the motors parameters are given in Table 7.5.

Generally, high gear ratios simplify the overall dynamics to that of the actuators [186]. In our particular case this simplifying assumption cannot apply due to the very low inertia values of the head's motors (coreless dc motors), which yield products $r_i^2 J_{mi}$ of the same magnitude or even smaller than some elements of $D(q)$.

Label	***Parameters***	***Motor1***	***Motor2***	***Unit***
J_{mi}	rotor inertia	0.8 10^{-7}	0.8 10^{-7}	$kg.m^2$
B_{mi}	viscous damping constant	0.5 10^{-7}	0.5 10^{-7}	Nms
K_{bi}	Back-EMF constant	1.65 10^{-3}	1.65 10^{-3}	$^{Vs}/_{rad}$
K_{mi}	motor torque constant	15.8 10^{-3}	15.8 10^{-3}	Nm/A
$\frac{R_i}{K_{mi}^2}$	motor regulation	80	80	$^{10^3}/_{Nms}$

Table 7.5: Head's motors parameters

However, the system being under feedback with a PD controller whose proportional and Derivative gains are respectively $K_p = 40$ and $K_D = 80$, according to equation (6.31), the closed loop dynamics can be written as

$$\overline{M}\ddot{\mathbf{q}} + \overline{N}\dot{\mathbf{q}} + \overline{\Phi} = \mathbf{q}^*_{lsp} \tag{7.9}$$

with matrix $\overline{M} = K_p^{-1}\left[D(\mathbf{q}) + J\right]$, matrix $\overline{N} = K_p^{-1}\left[C(\mathbf{q}, \dot{\mathbf{q}}) + B + F_v + K_D\right]$ and the vector $\overline{\Phi} = K_p^{-1}\left[G(\mathbf{q}) + F_c \tanh(\alpha\dot{\mathbf{q}}) + T_d + K_p\mathbf{q}\right]$.

Note that the inertia properties, the viscous damping and the friction introduced by the gearboxes are not known. It is therefore necessary to identify the overall system's parameters in order to obtain a model good enough to apply the compute-torque control scheme presented in Section 6.3.

7.2.1.2 Head's Closed loop Identification and Validation

Experimentally, the identification was performed with the *System identification toolbox* of Matlab. The overall closed loop dynamics of the head can be approximated by the following linear decoupled system

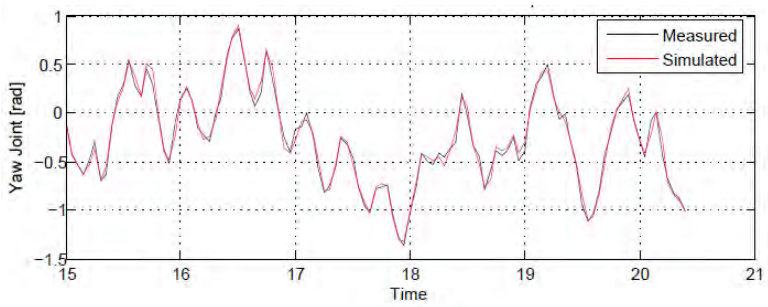
$$\begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix}\ddot{\mathbf{q}} + \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}\dot{\mathbf{q}} + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \mathbf{q}^*_{lsp} \tag{7.10}$$

where the identified closed loop parameters a_i, b_i and c_i are given in Table 7.6.

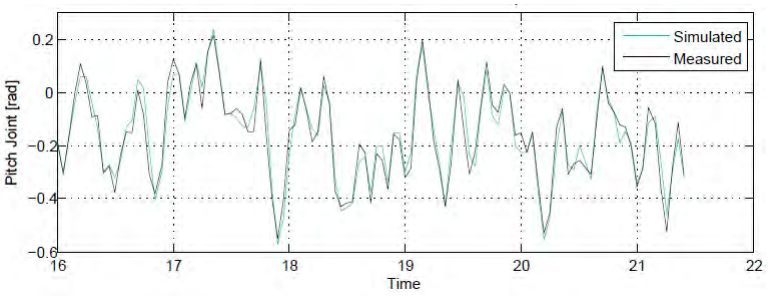
Label	***Value***	***Label***	***Value***
a_1	1.103474 10^{-3}	a_2	1.064349 10^{-3}
b_1	37.573973 10^{-3}	b_2	32.040737 10^{-3}
c_1	1.00172	c_2	1.000827

Table 7.6: Identified parameters of the approximated Head's closed loop

To validate this model, comparison of the model response with that of the actual system to given



(a) Yaw joint



(b) Pitch joint

Figure 7.12: Simulated responses of Head model

input signals covering the whole operating range is shown in Figure 7.12. It can be seen that the model reproduces fairly well the response of the actual system, confirming that it has captured the head's dynamics.

7.2.1.3 Controller Design

Following the design method in Section 6.6, the controller will have the form

$$\begin{aligned}\mathbf{q}^*_{lsp} &= \widehat{\overline{\mathrm{M}}}\hat{\ddot{\mathbf{q}}}_r + \widehat{\overline{\mathrm{N}}}\dot{\mathbf{q}}_r + \widehat{\overline{\Phi}} + \mu \\ &= \begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix} \hat{\ddot{\mathbf{q}}}_r + \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix} \dot{\mathbf{q}}_r + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + \mu\end{aligned}$$

with $\hat{\ddot{\mathbf{q}}}_r = \ddot{\mathbf{q}}_r - \alpha_v(\dot{\mathbf{q}} - \dot{\mathbf{q}}_r)$ and the reference trajectories $\dot{\mathbf{q}}_r$ and $\ddot{\mathbf{q}}_r$ given by equations (6.84) and (6.85) respectively.

$$\begin{cases} \dot{\mathbf{q}}_r &= -\Lambda_v \widehat{\mathrm{J}}_e^{\dagger} \left[sI + \Lambda_v\right]^{-1} \left(\Lambda_p e(t) - \widehat{\mathrm{L}}_e V_{off}\right) \\ \ddot{\mathbf{q}}_r &= -\Lambda_v \widehat{\mathrm{J}}_e^{\dagger} \left(\Lambda_p e(t) - \widehat{\mathrm{L}}_e V_{off}\right) - (\Lambda_v + \widehat{\mathrm{J}}_e^{\dagger}\dot{\mathrm{J}}_e)\dot{\mathbf{q}}_r \end{cases} \tag{7.11}$$

The auxiliary input μ and the feed-forward control input V_{off} are computed according to equations (6.133) and (6.149).

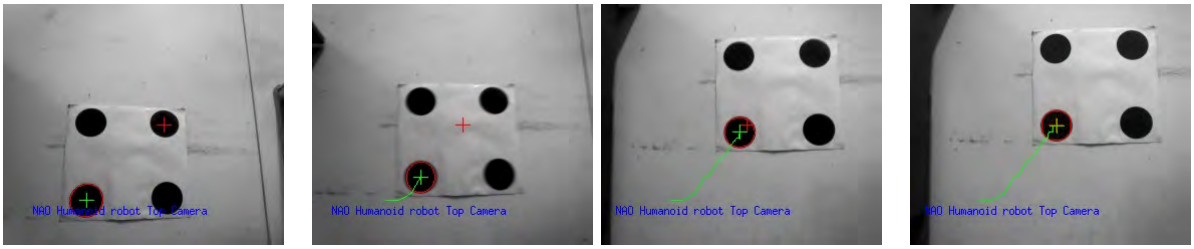
The proportional gain and the filter pole were respectively chosen as $\Lambda_p = 2.0$ and $\Lambda_v = 4.0$.

The matrix Q_s of the Lyapunov equation $A^T P_s + P_s A = -Q_s$ was set to a unit diagonal matrix, while the gain α_v was set to 10.

7.2.1.4 Application to Positioning

The target object chosen to validate this task as well as the tracking is a black dot on a white background. Thus, the positioning experiment consists of centering, from any initial position, a target point in the image.

In Figure 7.13 are shown four images representing respectively the initial, two intermediate and the final image configurations of the point. The center of gravity is indicated by a cross (green for the current and red for the desired).



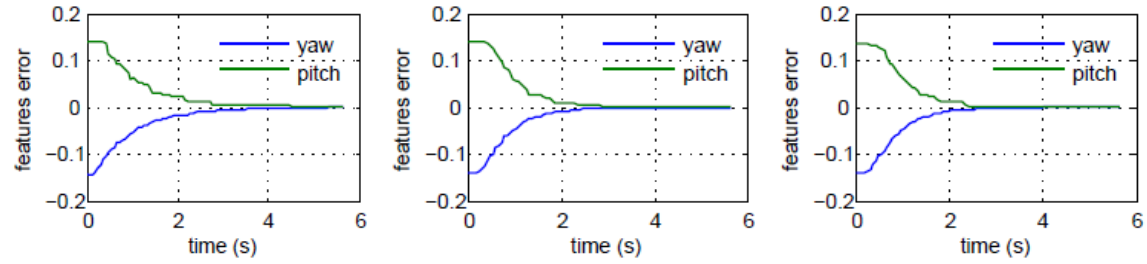
(a) Initial Configuration (b) Intermediate Configurations (c) Desired Configuration

Figure 7.13: Image configurations corresponding to experimental dynamic visual positioning task

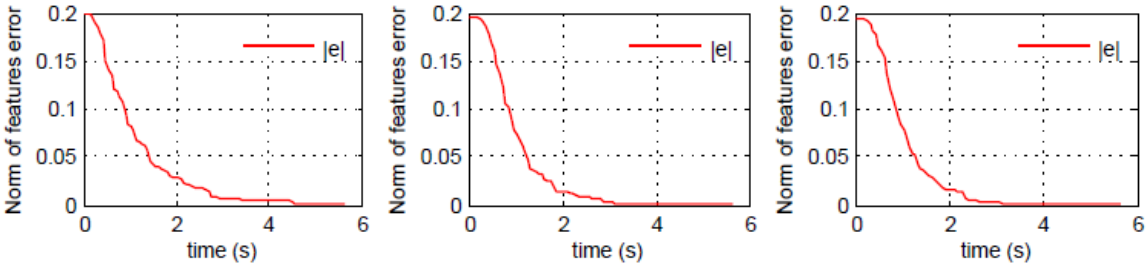
Without dynamic compensation, the task function, the command velocities and accelerations, and the joints angles under the classical, FE and FV laws are shown in Figure 7.14. As expected, it can be observed that the FE and the FV laws outperform the classical law. This experimentally validates the results obtained in simulations.

Applying now dynamic compensation, with the auxiliary input μ given by equation (6.133) or (6.138) approximated by $\mu = -\sigma \Lambda_p \mathbf{J}_e^T \boldsymbol{e}$, with $\sigma = 0.1$. The result of this experiment showing the task function, the joints velocities and accelerations are grouped in Figure 7.15.

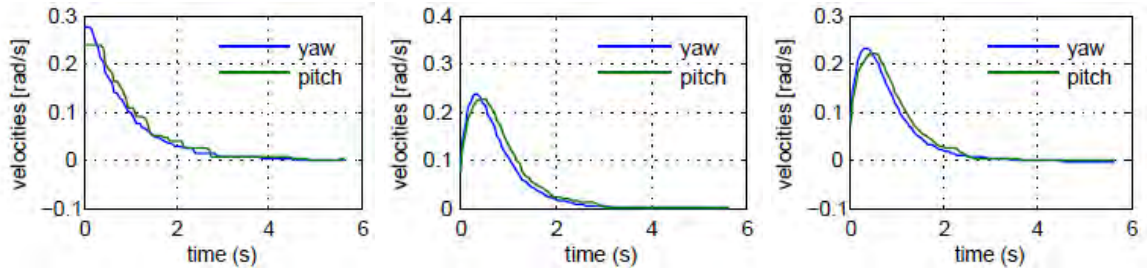
It can be observed a much faster response, which could be explained by the direct contribution of μ to the input signal, and $\alpha_v(\dot{q}_r - \dot{q})$ in the reference acceleration. When α_v increases, the tracking error decreases, we have noted however that the overall system response was slowing down. Thus, we have kept α_v to 10.



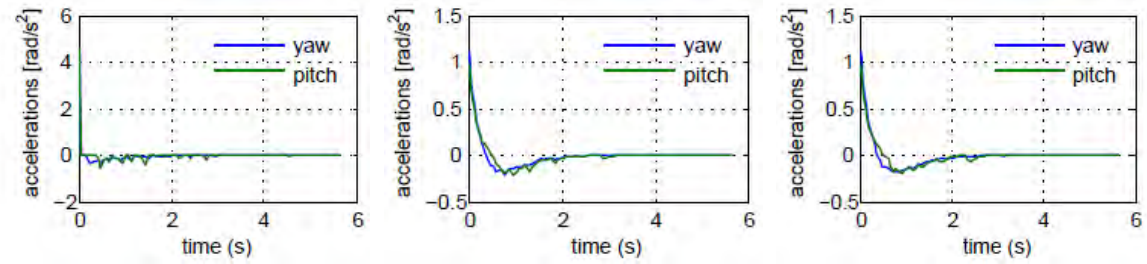
(a) Features error Classical law (b) Features error FEL (c) Features error FVL



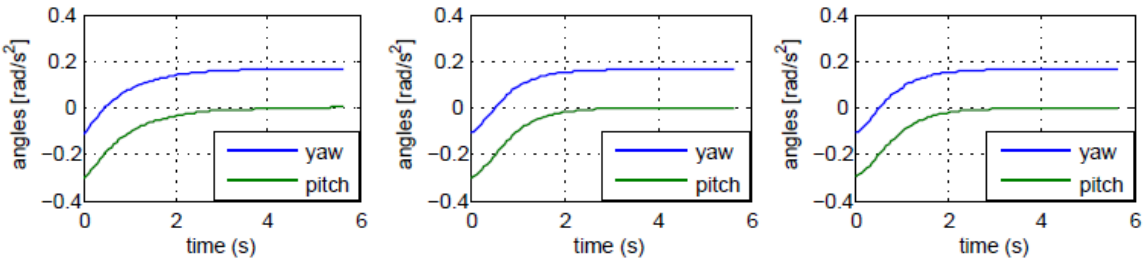
(d) Norm of error Classical law (e) Norm of error FE law (f) Norm of error FV law



(g) Velocities Classical law (h) Velocities FE law (i) Velocities FV law

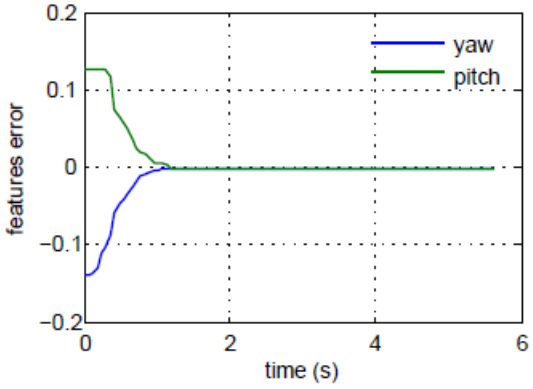


(j) Accelerations Classical law (k) Accelerations FE law (l) Accelerations FV law

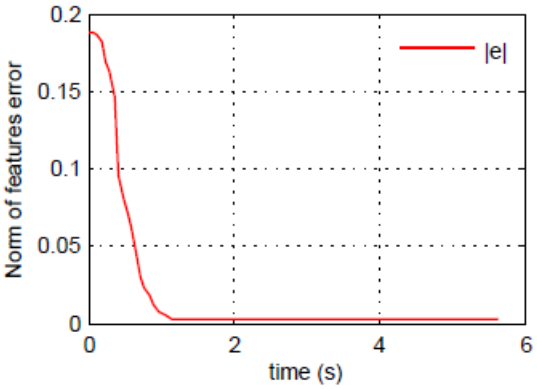


(m) Angular position Classical law (n) Angular position FE law (o) Angular position FV law

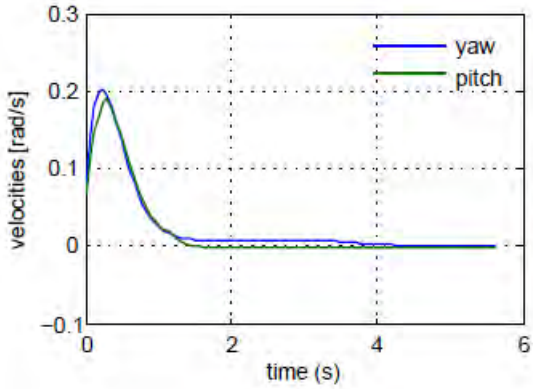
Figure 7.14: Experimental comparison between classical and proposed laws



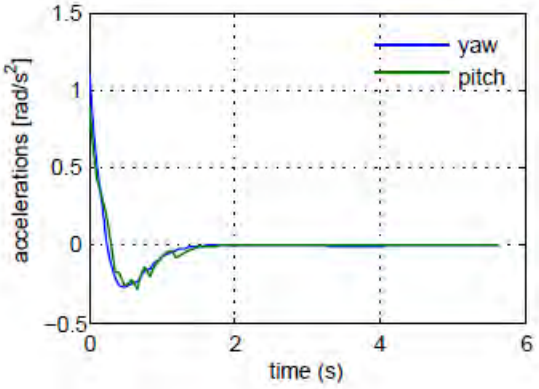
(a) Feature error



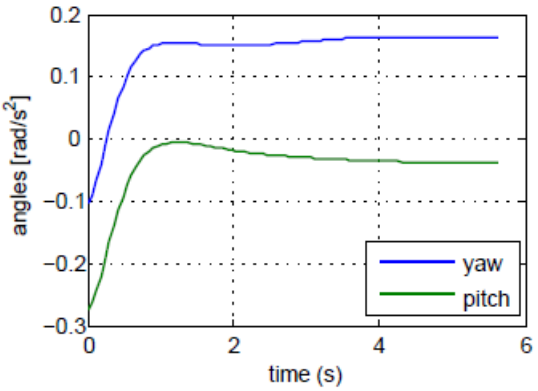
(b) Norm of feature error



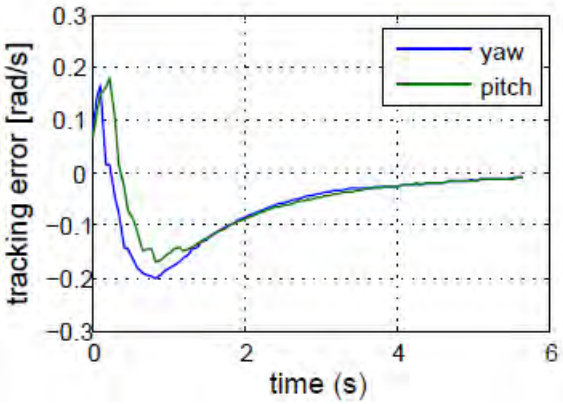
(c) Joints velocities



(d) Joints accelerations



(e) Joints angles



(f) Tracking

Figure 7.15: Positioning with dynamic compensation

7.2.1.5 Application to Tracking

Three tracking tasks were considered to evaluate and validate our compensation scheme. The first task consists of centering in the image a moving target point. The second task consists of tracking a moving visual set-point with the image of a fixed target point. And finally, in the third experiment, the target point and the visual set-point are both moving and the task consists of matching them.

The experimental setup is shown in Figure 7.16. Here again, a Kalman filter is used in the estimation of the target motion for compensation.

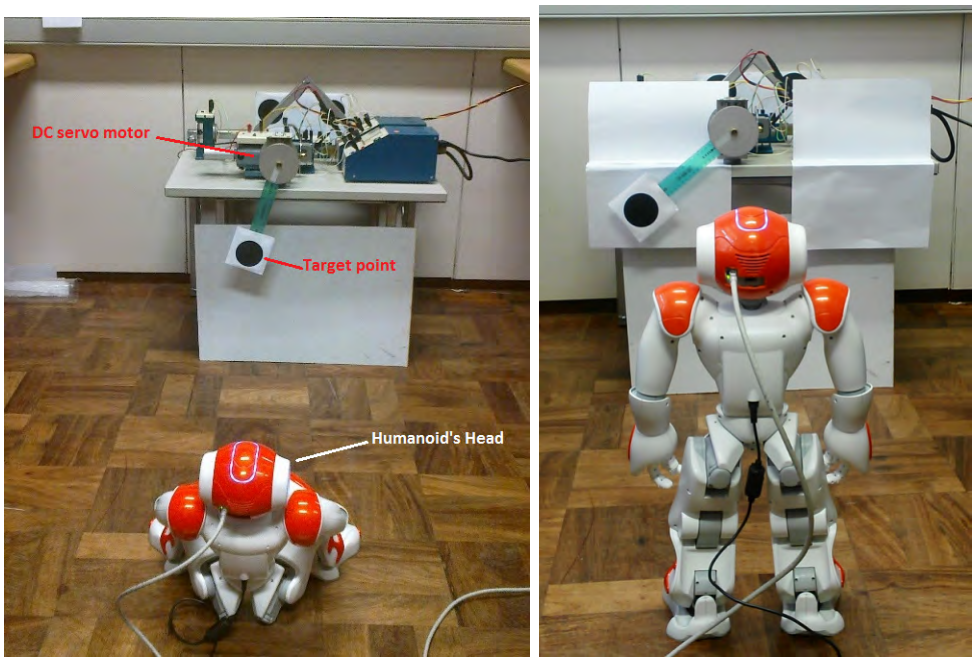
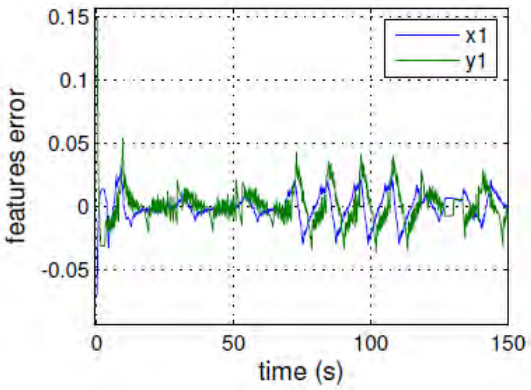


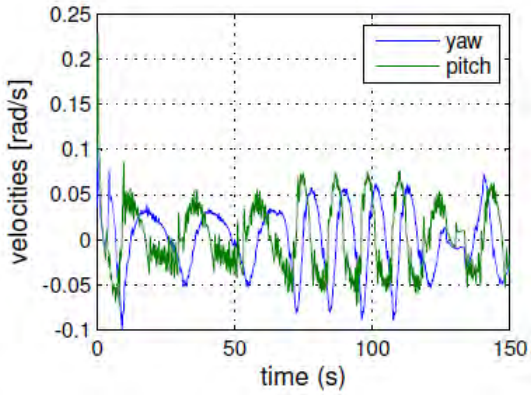
Figure 7.16: Experimental Setup for Dynamic visual tracking

Moving Target and Fixed Set-point. The results of the first experiment, where the target describes a circular trajectory is shown in Figure 7.17. There can be seen the image tracking error, the neck's joints angles, reference velocities and accelerations, and the image feature trajectory.

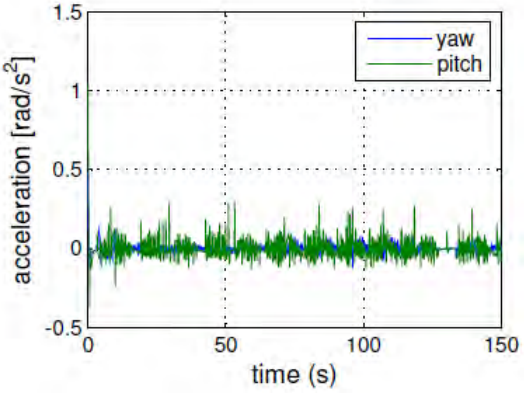
It can be noticed that the error increases with the target velocity. This could be explained by the error on the estimate of the target's motion, which is extracted from delayed images and, particularly, based on a model assuming constant acceleration with colored noise; whereas in the circular target's motion, the acceleration continuously varies and its magnitude depends on the angular velocity.



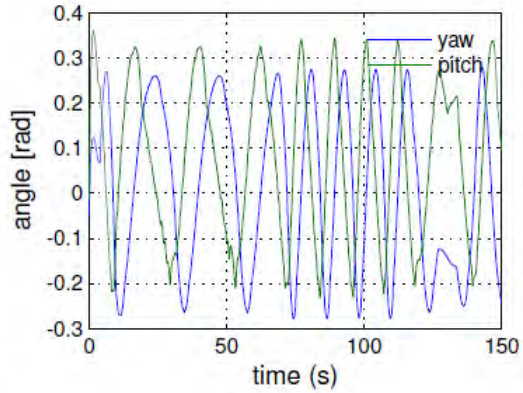
(a) Features error



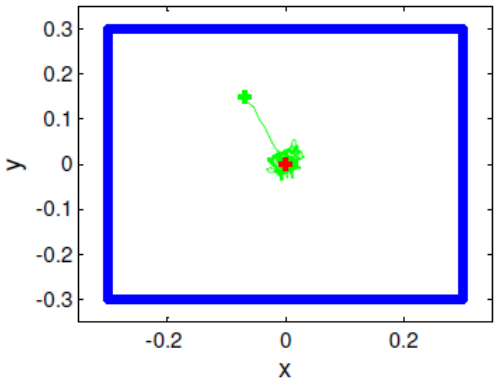
(b) Velocities of neck's joints



(c) Accelerations of neck's joints



(d) Neck's joints angles



(e) Image features trajectory

Figure 7.17: Tracking Circular trajectory motion

Fixed Target and Varying Set-point. The results of the second experiment, where the desired feature point describes a Lissajous curve such that

$$\begin{cases} x & = r\cos(\omega t) \\ y & = r\sin(2\omega t) \end{cases} \tag{7.12}$$

with $\omega = \frac{2\pi}{40}$, are shown in Figure 7.19.

In this experiment, the estimate of the target's motion in image $\widehat{\mathbf{L}}_e{}^c\widehat{V}_{o_{ff}}$ was applied in the filter as a variable structure control input as follows

$$\widehat{\mathbf{L}}_e{}^c\widehat{V}_{o_{ff}} = \begin{cases} k\left\|\widehat{\frac{\partial e}{\partial t}}\right\| \frac{e}{\|e\|} & ; if\ \|e\| > \varepsilon_o \\ k\left\|\widehat{\frac{\partial e}{\partial t}}\right\| \frac{e}{\varepsilon_o} & ; if\ \|e\| \leq \varepsilon_o \end{cases} \tag{7.13}$$

with $\varepsilon_o = 10^{-7}$ and the gain $k = 0.45$.

In Figure 7.18, we show a sequence of three images with the trajectories of the target point in green, while the desired point is indicated with a red cross.

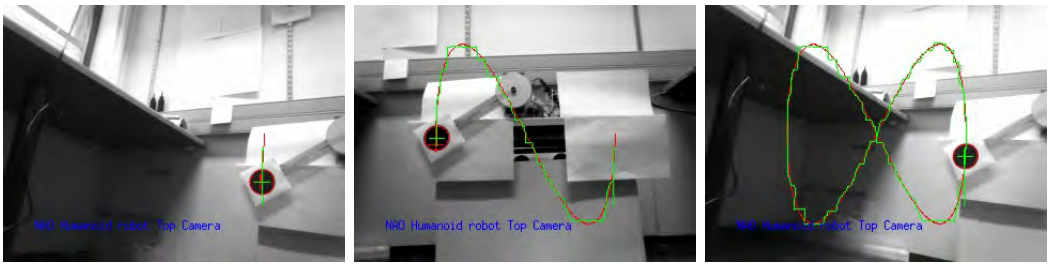
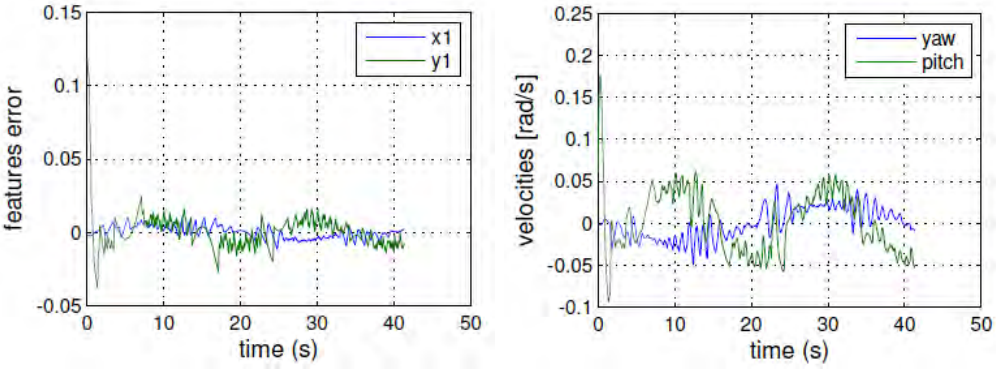


Figure 7.18: Sequence of image configurations while tracking a Lissajous figure

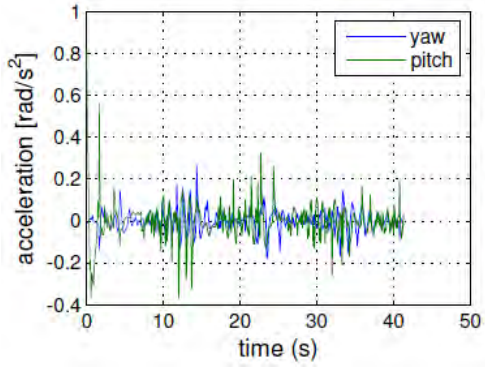
The convergence analysis of this scheme is given in Appendix C.2. This scheme yields small tracking error though with more oscillations on the reference commands, as shown, respectively, in Figure 7.19a, and Figures 7.19b and 7.19c.

These oscillations can be explained by the fact NAO's pitch joint is stiffer than the yaw joint, thus its motion as can be seen in Figure 7.18 on the vertical feature, evolves in a discrete manner. Additionally, the delayed estimate of the target motion being applied by the feed-forward control input with the same magnitude ($k\left\|\widehat{\frac{\partial e}{\partial t}}\right\|$) to all components of the features error, yields successive overshoots about the boundary layer ε_o.

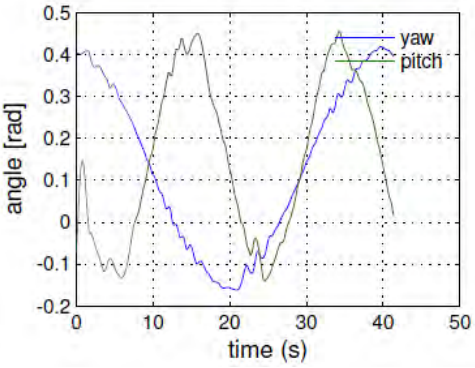


(a) Features error

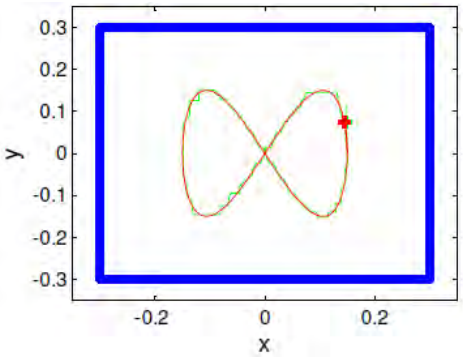
(b) Reference velocities



(c) Reference acceleration



(d) Angular positions



(e) Image features trajectory

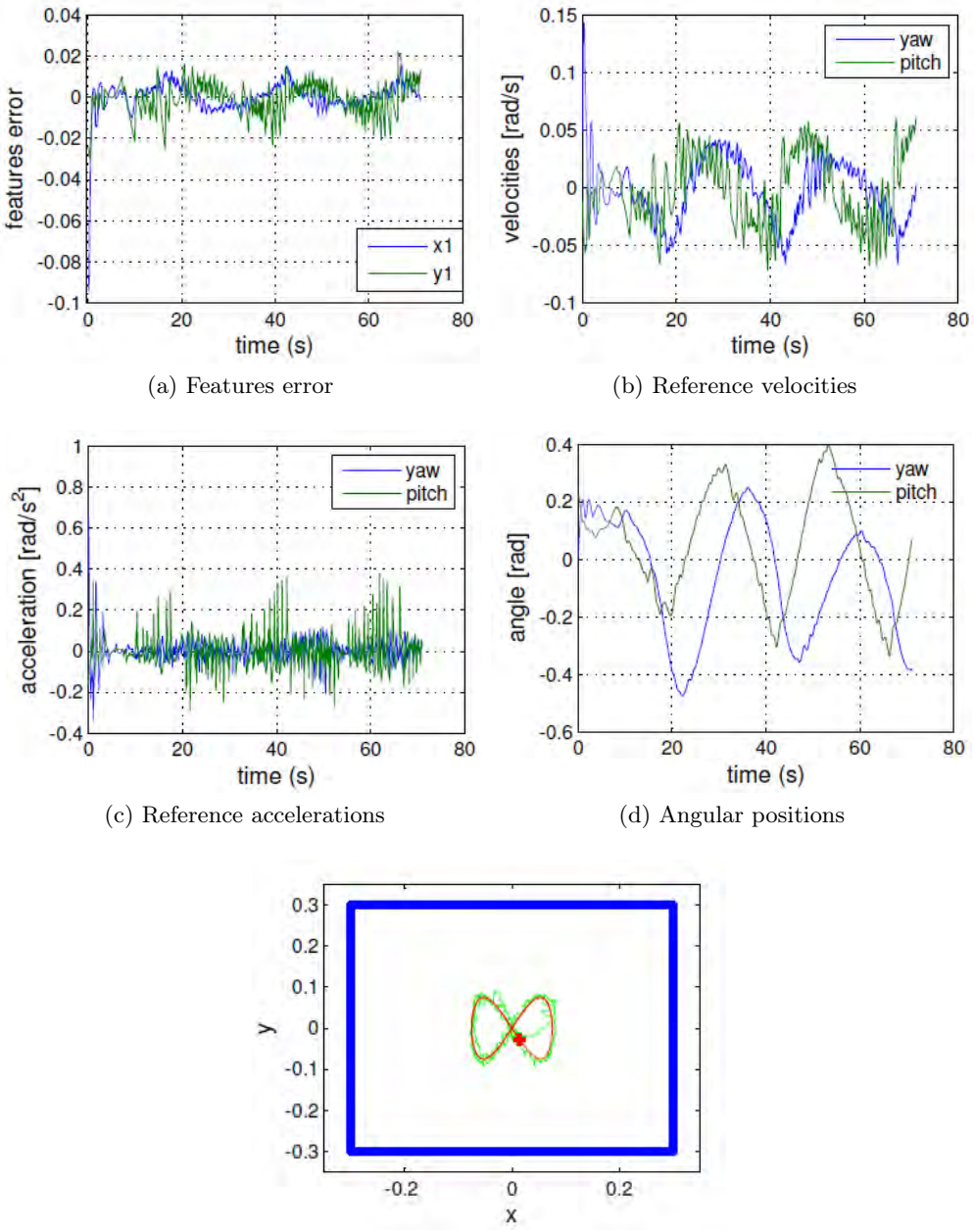
Figure 7.19: Tracking of a Lissajous figure

Moving Target and Varying Set-point. Finally, the results of the third experiment combining varying set-point and moving target are shown in Figure 7.20.

Here as well, the variable structure compensation of the target motion is used. Despite the complexity of the motion, it can be seen that the robot manages fairly well to match the target point to the desired feature points. This is confirmed by the magnitude of features error which stays relatively small (Figure 7.20a), though some oscillations are perceptible. Similarly, on the

reference commands plots (Figure 7.20b and Figure 7.20c) the oscillations are also perceptible.

The image features trajectory is shown in Figure 7.20e, and a sequence of nine images corresponding to different configuration during the tracking is given in Figure 7.21.



(a) Features error

(b) Reference velocities

(c) Reference accelerations

(d) Angular positions

(e) Image feature trajectories

Figure 7.20: Tracking moving target and varying set-point

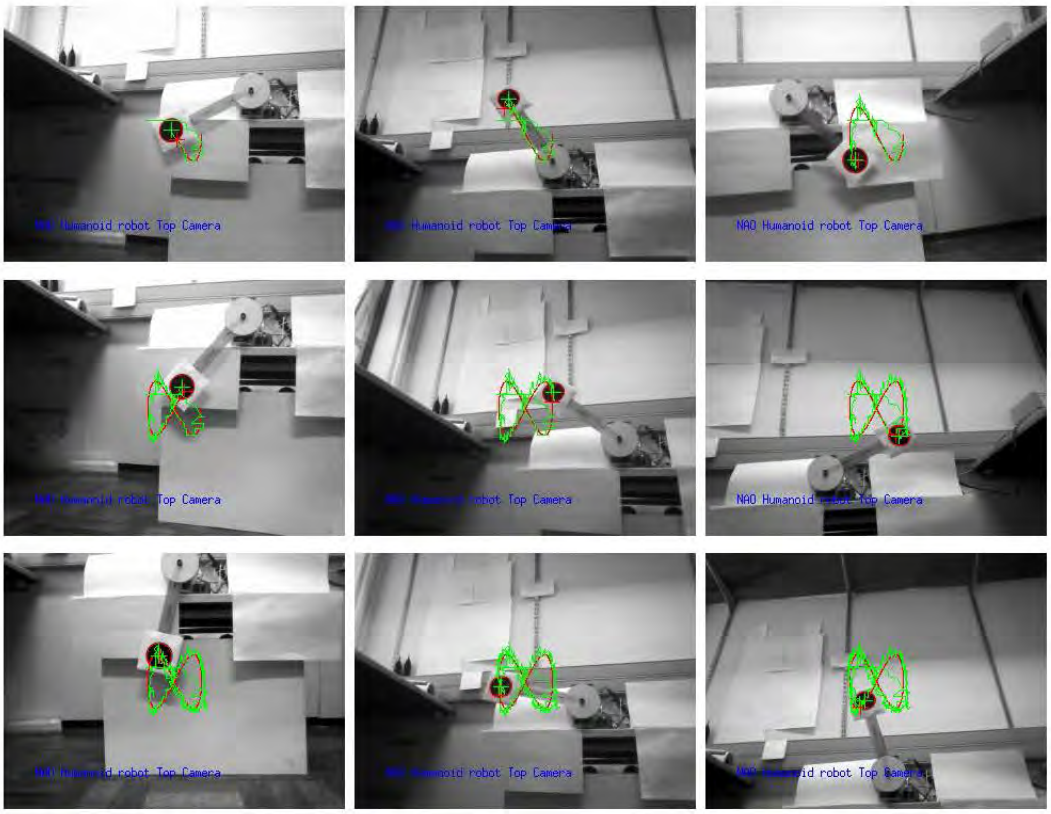


Figure 7.21: Sequence of image configurations while tracking moving target with varying set-point

7.3 Conclusion

In this chapter, we have validated the proposed approach on dynamic visual servoing. The two new laws derived in the previous chapter were compared to the classical visual servoing law. Considering the speed of response and the maximum control action required as evaluation criteria, we showed, for different cases, through extensive simulations and experiments that the proposed control laws outperform the classical one.

The dynamic compensation scheme of the vision and robot dynamics was validated on a two degrees of freedom humanoid robot's head. To illustrate the effectiveness of our design, we considered positioning as well as tracking tasks with constant and varying references of both the target in Cartesian space and its desired position in the image space. The target motion and any other disturbance were estimated with a Kalman filter and more effectively compensated with a variable structure feed-forward control input. Despite the effect of the delay, not considered in our formulation and slightly perceptible in the tracking, the proposed controller kept the error relatively small.

Chapter 8

General Conclusion

The work we have presented has attempted to answer the theoretical and practical problem stated in the introduction. The thesis was mainly about reactive control by visual servoing of humanoid robots in order to give them a degree of autonomy in performing some tasks such as positioning with respect to an object, tracking, and grasping an object. The work has also explored dynamic aspects in visual servoing, namely the improvement of reference commands issued by the visual controller and the robot's tracking performances for these reference trajectories. Thus, this work has brought the following contributions

Locomotion control: After deriving the dynamic model of a humanoid robot and discussing their control problem, which reduces to generating trajectories specifying how the biped robot should move to realize a stable gait, we have presented the concept of the ZMP and introduced LMPC based walking pattern generators using simplified dynamic models of humanoid robot. We particularly focused on the reactive pattern generator proposed by Herdt et al.,[23, 27], able to generate automatically the footsteps positions from reference velocities. As that pattern generator is only reactive for translations, we proposed an extension for orientations to overcome their predetermination which was previously required. We therefore obtained a reactive omnidirectional walking pattern generator. We additionally proposed dynamic planning algorithm for the swing foot trajectories and validated the walking pattern generator in simulation.

Visual Servoing on Humanoid Robot: As a way to determine a task Jacobian compatible with the humanoid bipedal structure and the walking constraints, we proposed a reformulation of the kinematics of the humanoid robot's active chain (stance foot, torso and either the head where the camera is located or the hand where the gripper is located) using the floating-base concept. In this manner continuous dynamics could be separated from the discrete dynamics of this hybrid system, allowing thus to define easily a continuous task Jacobian and to integrate locomotion and its related constraints. This formulation offers the advantage of being general, it is not limited to biped robots. It can be applied to multi-legged robots or even to crawling robots, for example. The redundancy problem had not been forgotten, it has been addressed within a

task priority framework, where the gradient projective method was used to realize auxiliary tasks such as the alignment of the humanoid's body and sight direction, joint limits avoidance, and the improvement of manipulability.

Dynamic visual servoing: After reviewing and discussing existing dynamic visual servoing approaches, we considered approaches accounting for the nonlinear dynamic of the robot and considered a dynamic control at two levels. On the one hand, from classical visual servoing laws, we proposed two new laws allowing second order decrease of the features error. This yields faster and damped image responses which require a maximal velocity less than that of classical visual servoing laws. In addition to reference velocity, the reference acceleration can be generated without the need to derive or approximate the optical flow. Using perturbation theory, we analyzed the conditions of stability and convergence of the proposed laws. On the other hand, we have studied the dynamic interaction between the vision and the robotic subsystems. Then, using Lyapunov direct method, we designed a robust adaptive controller for the overall system and proved the ultimate uniform boundedness of the close-loop solution.

Experimentally: We have tested all these theoretical considerations on the humanoid robot NAO. The results obtained confirmed the validity of our approach.

The solutions proposed in these thesis have numerous aspects that have not been researched in as much depth as we had wished. If the proposed control schemes have shown, to our view, the benefits of visual servoing in the control of humanoid robots, this work represents only a first step which needs further investigations.

Thus, future works could consider for example, the inclusion of the humanoid robot's full dynamics in the formulation of its visual servoing. This will allow the compensation of dynamic interactions between chains; for instance the one observed during grasping, where the robot drifted from its desired trajectory when the arm was stretched. One could also consider a reactive walking pattern generator based on the capture point [139] instead of the ZMP. Indeed, such a formulation will be more robust since it would account for the angular momentum about the CoM, which have been neglected in this study. The redundancy could be handled within a more general framework, such as task sequencing, allowing more auxiliary objectives to be considered, for instance obstacles avoidance, maintaining features visibility and more.

In order to increase the autonomy, one may consider combining visual servoing with an object recognition algorithm which could detect the target and assign automatically the image features instead of assigning them manually. This could further be integrated in a more complex task where sub-objectives are realized by visual servoing.

List of Publications

Bombile, Michael. "Visual Servoing Based Positioning and Object Tracking on Humanoid Robot." In New Trends in Networking, Computing, E-learning, Systems Sciences, and Engineering, pp. 19-27. Springer International Publishing, 2015.

Bohra, P., and M. Bombile. "Towards a Passive Gait: Modeling the Fully Actuated Humanoid NAO." In New Trends in Networking, Computing, E-learning, Systems Sciences, and Engineering, pp. 505-512. Springer International Publishing, 2015.

Bibliography

[1] Aldebaran-robotics, "NAO Technical overview, Actuator and Sensor list." http://doc.aldebaran.com/2-1/family/nao-dcm/actuator-sensor-names.html, 2015. Retrieved March 25, 2015.

[2] L. E. Weiss, A. C. Sanderson, and C. P. Neuman, "Dynamic sensor-based control of robots with visual feedback," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 5, pp. 404–417, 1987.

[3] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *Robotics Automation Magazine, IEEE*, vol. 13, no. 4, pp. 82–90, 2006.

[4] G. Taylor, L. Kleeman, *et al.*, "Flexible self-calibrated visual servoing for a humanoid robot," in *Australian Conference on Robotics and Automation (ACRA2001)*, pp. 79–84, 2001.

[5] Y. Pang, Q. Huang, D. Jia, Y. Tian, J. Gao, and W. Zhang, "Object manipulation of a humanoid robot based on visual Servoing," *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1124–1129, Oct. 2007.

[6] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, and D. Kragic, "Grasping Known Objects with Humanoid Robots : A Box-Based Approach," *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pp. 1–6, 2009.

[7] K. Namba and N. Maru, "3D linear visual servoing for humanoid robot," *IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02*, vol. 3, pp. 2225–2230, 2002.

[8] S. Mukai and N. Maru, "Redundant Arm Control by Linear Visual Servoing Using Pseudo Inverse Matrix," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1237–1242, Oct. 2006.

[9] Y. Shibuya and N. Maru, "Control of 6 dof arm of the humanoid robot by linear visual servoing," in *Industrial Electronics, 2009. ISIE 2009. IEEE International Symposium on*, pp. 1791–1796, July 2009.

[10] A. A. Moughlbay, E. Cervera, and P. Martinet, "Error regulation strategies for Model Based visual servoing tasks: Application to autonomous object grasping with Nao robot,"

2012 12th International Conference on Control Automation Robotics & Vision (ICARCV), pp. 1311–1316, Dec. 2012.

[11] D. J. Agravante, J. Pagès, and F. Chaumette, "Visual Servoing for the REEM Humanoid Robot's Upper Body," in *IEEE Int. Conf. on Robotics and Automation, ICRA'13*, (Karlsruhe, Allemagne), pp. 5233–5238, May 2013.

[12] J. Peng, A. Peters, X. Ao, and A. Srikaew, "Grasping a waving object for a humanoid robot using a biologically-inspired active vision system," in *Robot and Human Interactive Communication, 2003. Proceedings. ROMAN 2003. The 12th IEEE International Workshop on*, pp. 115–120, Oct. 2003.

[13] A. Moughlbay, Amine, E. Cervera, and M. Philippe, "Real-time model based visual servoing tasks on a humanoid robot," in *Intelligent Autonomous Systems 12*, Advances in Intelligent Systems and Computing, pp. pp 321–333, Springer Berlin Heidelberg, 2013.

[14] G. Taylor and L. Kleeman, "Hybrid position-based visual servoing with online calibration for a humanoid robot," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, pp. 686–691, 2004.

[15] F. Chaumette and S. Hutchinson, "Visual servo control. ii. advanced approaches [tutorial]," *Robotics Automation Magazine, IEEE*, vol. 14, no. 1, pp. 109–118, 2007.

[16] P.-B. Wieber, *Modélisation et commande d'un robot marcheur anthropomorphe*. PhD thesis, École Nationale Supérieure des Mines de Paris, 2000.

[17] M. Vukobratović and J. Stepanenko, "On the stability of anthropomorphic systems," *Mathematical Biosciences*, vol. 15, no. 1, pp. 1–37, 1972.

[18] M. Vukobratovic and B. Borovac, "Zero-moment point - thirty five years of its life," *International Journal of Humanoid Robotics*, vol. 1, no. 01, pp. 157–173, 2004.

[19] C. Dune, a. Herdt, O. Stasse, P.-B. Wieber, K. Yokoi, and E. Yoshida, "Cancelling the sway motion of dynamic walking in visual servoing," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3175–3180, Oct. 2010.

[20] C. Dune, A. Herdt, E. Marchand, O. Stasse, P.-B. Wieber, and E. Yoshida, "Vision based control for Humanoid robots," in *IROS Workshop on Visual Control of Mobile Robots (ViCoMoR)*, (San Francisco, USA, États-Unis), pp. 19–26, 2011.

[21] M. Garcia, O. Stasse, J.-B. Hayet, C. Esteves, and J.-P. Laumond, "Vision-based motion primitives for reactive walking," in *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*, pp. 286–291, Oct 2013.

[22] M. Garcia, O. Stasse, J.-B. Hayet, C. Dune, C. Esteves, and J.-P. Laumond, "Vision-guided motion primitives for humanoid reactive walking: Decoupled versus coupled approaches," *The International Journal of Robotics Research*, p. 0278364914550891, 2014.

[23] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl, "Online walking motion generation with automatic footstep placement," *Advanced Robotics*, vol. 24, no. 5-6, pp. 719–737, 2010.

[24] A. Faragasso, G. Oriolo, A. Paolillo, and M. Vendittelli, "Vision-based corridor navigation for humanoid robots," *2013 IEEE International Conference on Robotics and Automation*, pp. 3190–3195, May 2013.

[25] N. Mansard, O. Stasse, F. Chaumette, and K. Yokoi, "Visually-guided grasping while walking on a humanoid robot," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 3041–3047, 2007.

[26] P. I. Corke and M. C. Good, "Dynamic effects in visual closed-loop systems," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 671–683, 1996.

[27] A. Herdt, N. Perrin, and P.-B. Wieber, "Walking without thinking about it," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 190–195, Oct. 2010.

[28] M. Garcia, O. Stasse, and J.-B. Hayet, "Vision-driven walking pattern generation for humanoid reactive walking," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 216–221, May 2014.

[29] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *The International Journal of Robotics Research*, vol. 6, no. 2, pp. 3–15, 1987.

[30] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *Robotics and Automation, IEEE Transactions on*, vol. 12, pp. 651–670, Oct. 1996.

[31] P. I. Corke *et al.*, *Visual Control of Robots: high-performance visual servoing*. Research Studies Press Taunton, UK, 1996.

[32] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation," tech. rep., COMPUTATIONAL VISION AND ACTIVE PERCEPTION LABORATORY, 2002.

[33] E. Malis, "Survey of vision-based robot control," *ENSIETA European Naval Ship Design Short Course, Brest, France*, 2002.

[34] M. A.-R. Marey, *Contribution to control modeling in visual servoing, task Redundancy, and joint limits avoidance*. PhD thesis, Universite de Rennes 1, 2010.

[35] M. L. B. C. Samson and B. Espiau, "Robot control :the task function approach, oxford university press, oxford, 1991," *Robotica*, vol. 9, pp. 447–448, Dec. 1991.

[36] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 313–326, 1992.

[37] L. E. Weiss, A. C. Sanderson, and C. Neuman, "Dynamic visual servo control of robots: an adaptive image-based approach," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, vol. 2, pp. 662–668, IEEE, 1985.

[38] M. Kazemi, K. Gupta, and M. Mehrandezh, "Path-planning for visual servoing: A review and issues," in *Visual Servoing via Advanced Numerical Methods*, pp. 189–207, Springer, 2010.

[39] J. Pomares, I. Perea, G. J. García, C. A. Jara, J. A. Corrales, and F. Torres, "A multisensorial hybrid control for robotic manipulation in human-robot workspaces," *Sensors*, vol. 11, no. 10, pp. 9839–9862, 2011.

[40] J. A. Gangloff, M. de Mathelin, and G. Abba, "6 dof high speed dynamic visual servoing using gpc controllers," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 3, pp. 2008–2013, IEEE, 1998.

[41] J. A. Gangloff and M. F. De Mathelin, "High speed visual servoing of a 6 dof manipulator using mimo predictive control," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 4, pp. 3751–3756, IEEE, 2000.

[42] R. Dahmouche, N. Andreff, Y. Mezouar, O. Ait-Aider, and P. Martinet, "Dynamic visual servoing from sequential regions of interest acquisition," *The International Journal of Robotics Research*, vol. 31, pp. 520–537, Feb. 2012.

[43] E. Malis, *Contributions à la modélisation et à la commande en asservissement visuel*. PhD thesis, 1998.

[44] E. Malis, F. Chaumette, and S. Boudet, "21/2d visual servoing," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 2, pp. 238–250, 1999.

[45] O. Faugeras, *Three-dimensional computer vision: a geometric viewpoint*. MIT press, 1993.

[46] G. Xu and Z. Zhang, *Epipolar geometry in stereo, motion and object recognition: a unified approach*, vol. 6. Springer, 1996.

[47] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[48] D. A. Forsyth and J. Ponce, *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[49] Z. Zhang, "On the epipolar geometry between two images with lens distortion," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 1, pp. 407–411, IEEE, 1996.

[50] L. Robert, "Camera calibration without feature extraction," *Computer Vision and Image Understanding*, vol. 63, no. 2, pp. 314–325, 1996.

[51] G. P. Stein, "Lens distortion calibration using point correspondences," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pp. 602–608, IEEE, 1997.

[52] A. De Luca, G. Oriolo, and P. R. Giordano, "Image-based visual servoing schemes for nonholonomic mobile manipulators," *Robotica*, vol. 25, pp. 131–145, Mar. 2007.

[53] E. Malis, "Visual servoing invariant to changes in camera-intrinsic parameters," *Robotics and Automation, IEEE Transactions on*, vol. 20, no. 1, pp. 72–81, 2004.

[54] M. Marey and F. Chaumette, "Analysis of classical and new visual servoing control laws," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 3244–3249, IEEE, 2008.

[55] W. J. Wilson, C. Williams Hulls, and G. S. Bell, "Relative end-effector control using cartesian position based visual servoing," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 5, pp. 684–696, 1996.

[56] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The confluence of vision and control*, pp. 66–78, Springer, 1998.

[57] P. Martinet and J. Gallice, "Position based visual servoing using a non-linear approach," in *Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on*, vol. 1, pp. 531–536, IEEE, 1999.

[58] R. M. Haralick, H. Joo, D. Lee, S. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose estimation from corresponding point data," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 19, no. 6, pp. 1426–1446, 1989.

[59] D. F. Dementhon and L. S. Davis, "Model-based object pose in 25 lines of code," *International journal of computer vision*, vol. 15, no. 1-2, pp. 123–141, 1995.

[60] M. Liu and K. Wong, "Pose estimation using four corresponding points," *Pattern Recognition Letters*, vol. 20, pp. 69–74, Jan. 1999.

[61] A. Ansar and K. Daniilidis, "Linear pose estimation from points or lines," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 5, pp. 578–589, 2003.

[62] G. Chesi, "Estimation of the camera pose from image point correspondences through the essential matrix and convex optimization," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 35–40, IEEE, 2009.

[63] F. Dornaika and C. Garcia, "Pose estimation using point and line correspondences," *Real-Time Imaging*, vol. 5, no. 3, pp. 215–230, 1999.

[64] J. E. McInroy and Z. Qi, "A novel pose estimation algorithm based on points to regions correspondence," *Journal of Mathematical Imaging and Vision*, vol. 30, no. 2, pp. 195–207, 2008.

[65] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, MA Fischler and O. Firschein, eds*, pp. 61–62, 1987.

[66] E. Malis and F. Chaumette, "2 1/2 d visual servoing with respect to unknown objects through a new estimation scheme of camera displacement," *International Journal of Computer Vision*, vol. 37, no. 1, pp. 79–97, 2000.

[67] E. Malis, F. Chaumette, and S. Boudet, "2d 1/2 visual servoing stability analysis with respect to camera calibration errors," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, vol. 2, pp. 691–697 vol.2, 1998.

[68] C. Perez-Vidal, L. Gracia, N. Garcia, and E. Cervera, "Visual Control of Robots with Delayed Images," *Advanced Robotics*, vol. 23, pp. 725–745, Jan. 2009.

[69] J. T. Feddema, C. G. Lee, and O. R. Mitchell, "Weighted selection of image features for resolved rate visual feedback control," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 1, pp. 31–47, 1991.

[70] K. Hashimoto and T. Noritsugu, "Potential problems and switching control for visual servoing," in *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, vol. 1, pp. 423–428, IEEE, 2000.

[71] F. Chaumette, *La relation vision-commande: théorie et application à des taches robotiques.* PhD thesis, Université de Rennes 1, 1990.

[72] F. Chaumette and A. Modeling, "Image Moments : A General and Useful Set of Features for Visual Servoing," vol. 20, no. 4, pp. 713–723, 2004.

[73] O. Tahri and F. Chaumette, "Point-based and region-based image moments for visual servoing of planar objects," *IEEE Transactions on Robotics*, vol. 21, pp. 1116–1127, Dec. 2005.

[74] L. Matthies, T. Kanade, and R. Szeliski, "Kalman filter-based algorithms for estimating depth from image sequences," *International Journal of Computer Vision*, vol. 3, no. 3, pp. 209–238, 1989.

[75] a. De Luca, G. Oriolo, and P. Robuffo Giordano, "Feature Depth Observation for Image-based Visual Servoing: Theory and Experiments," *The International Journal of Robotics Research*, vol. 27, pp. 1093–1116, Oct. 2008.

[76] A. P. Dani and W. E. Dixon, "Single camera structure and motion estimation," in *Visual Servoing via Advanced Numerical Methods*, pp. 209–229, Springer, 2010.

[77] H. Michel, P. Rives, *et al.*, "Singularities in the determination of the situation of a robot effector from the perspective view of 3 points," *Research report n1850, INRIA*, vol. - Research report, February, 1993.

[78] G. Chesi, Y. S. Hung, and H. L. Yung, "Image measurement errors in visual servoing: Estimating the induced positioning error," in *Visual Servoing via Advanced Numerical Methods*, pp. 151–167, Springer, 2010.

[79] B. Espiau, "Effect of camera calibration errors on visual servoing in robotics," in *Experimental Robotics III*, pp. 182–192, Springer, 1994.

[80] G. D. Hager, W.-C. Chang, and A. S. Morse, "Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pp. 2850–2856, IEEE, 1994.

[81] G. D. Hager, "A modular system for robust positioning using feedback from stereo vision," *Robotics and Automation, IEEE Transactions on*, vol. 13, no. 4, pp. 582–595, 1997.

[82] M. Iwatsuki and N. Okiyama, "A new formulation of visual servoing based on cylindrical coordinate system," *Robotics, IEEE Transactions on*, vol. 21, no. 2, pp. 266–273, 2005.

[83] P. I. Corke, F. Spindler, and F. Chaumette, "Combining Cartesian and polar coordinates in IBVS," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5962–5967, Oct. 2009.

[84] O. Tahri, F. Chaumette, and Y. Mezouar, "New decoupled visual servoing scheme based on invariants from projection onto a sphere," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 3238–3243, 2008.

[85] O. Tahri, Y. Mezouar, F. Chaumette, and P. Corke, "Generic decoupled image-based visual servoing for cameras obeying the unified projection model," *2009 IEEE International Conference on Robotics and Automation*, pp. 1116–1121, May 2009.

[86] O. Tahri, Y. Mezouar, F. Chaumette, and P. Corke, "Generic decoupled image-based visual servoing for cameras obeying the unified projection model," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 1116–1121, 2010.

[87] R. T. Fomena, O. Tahri, and F. X. E. O. Chaumette, "Distance-Based and Orientation-Based Visual Servoing From Three Points," *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 256–267, 2011.

[88] K. Deguchi, "Optimal motion control for image-based visual servoing by decoupling translation and rotation," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, vol. 2, pp. 705–711, IEEE, 1998.

[89] F. Chaumette and E. Malis, "2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings," *Proceedings 2000 ICRA Millennium Conference IEEE International Conference on Robotics and Automation Symposia Proceedings Cat No00CH37065*, vol. 1, no. April, pp. 630–635, 2000.

[90] P. I. Corke and S. A. Hutchinson, "A new hybrid image-based visual servo control scheme," in *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, vol. 3, pp. 2521–2526, IEEE, 2000.

[91] P. I. Corke and S. A. Hutchinson, "A new partitioned approach to image-based visual servo control," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 4, pp. 507–515, 2001.

[92] N. R. Gans and S. A. Hutchinson, "An asymptotically stable switched system visual controller for eye in hand robots," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 1, pp. 735–742, IEEE, 2003.

[93] N. R. Gans, S. a. Hutchinson, and P. I. Corke, "Performance Tests for Visual Servo Control Systems, with Application to Partitioned Approaches to Visual Servo Control," *The International Journal of Robotics Research*, vol. 22, pp. 955–981, Oct. 2003.

[94] N. R. Gans and S. A. Hutchinson, "Stable visual servoing through hybrid switched-system control," *Robotics, IEEE Transactions on*, vol. 23, no. 3, pp. 530–540, 2007.

[95] O. Kermorgant and F. Chaumette, "Combining ibvs and pbvs to ensure the visibility constraint," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 2849–2854, IEEE, 2011.

[96] L. Deng, F. Janabi-Sharifi, and W. J. Wilson, "Hybrid motion control and planning strategies for visual servoing," *Industrial Electronics, IEEE Transactions on*, vol. 52, no. 4, pp. 1024–1040, 2005.

[97] A. Crétual and F. Chaumette, "Visual servoing based on image motion," *The International Journal of Robotics Research*, vol. 20, no. 11, pp. 857–877, 2001.

[98] Y. Mezouar and F. Chaumette, "Path planning for robust image-based control," *IEEE Transactions on Robotics and Automation*, vol. 18, pp. 534–549, Aug. 2002.

[99] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[100] N. Cowan and D. Koditschek, "Planar image based visual servoing as a navigation problem," *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 1, pp. 611–617, 1999.

[101] N. Cowan, J. Weingarten, and D. Koditschek, "Visual servoing via navigation functions," *IEEE Transactions on Robotics and Automation*, vol. 18, pp. 521–533, Aug. 2002.

[102] F. Schramm, A. Micaelli, and G. Morel, "Calibration free path planning for visual servoing yielding straight line behaviour both in image and work space," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pp. 2216–2221, IEEE, 2005.

[103] F. Schramm and G. Morel, "Ensuring visibility in calibration-free path planning for image-based visual servoing," *Robotics, IEEE Transactions on*, vol. 22, no. 4, pp. 848–854, 2006.

[104] F. Schramm, F. Geffard, G. Morel, and A. Micaelli, "Calibration free image point path planning simultaneously ensuring visibility and controlling camera path," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 2074–2079, IEEE, 2007.

[105] G. Chesi, "Visual servoing path planning via homogeneous forms and lmi optimizations," *Robotics, IEEE Transactions on*, vol. 25, no. 2, pp. 281–291, 2009.

[106] K. Huebner and D. Kragic, "Selection of robot pre-grasps using box-based shape approximation," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 1765–1770, IEEE, 2008.

[107] D. J. Agravante, A. Cherubini, A. Bussy, and A. Kheddar, "Human-humanoid joint haptic table carrying task with height stabilization using vision," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 4609–4614, IEEE, 2013.

[108] N. Hogan, "Impedance control: An approach to manipulation," in *American Control Conference, 1984*, pp. 304–313, IEEE, 1984.

[109] N. Vahrenkamp, S. Wieland, P. Azad, D. Gonzalez, T. Asfour, and R. Dillmann, "Visual servoing for humanoid grasping and manipulation tasks," *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*, pp. 406–412, Dec. 2008.

[110] M. Bishay, R. A. Peters, D. M. Wilkes, and K. Kawamura, "Hand-eye coordination with an active camera head," in *Intelligent Systems & Advanced Manufacturing*, pp. 406–417, International Society for Optics and Photonics, 1997.

[111] J. W. Grizzle, C. Chevallereau, R. W. Sinnet, and A. D. Ames, "Models, feedback control, and open problems of 3d bipedal robotic walking," *Automatica*, 2014.

[112] S. Czarnetzki, S. Kerner, and O. Urbann, "Observer-based dynamic walking control for biped robots," *Robotics and Autonomous Systems*, vol. 57, no. 8, pp. 839–845, 2009.

[113] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, pp. 1620–1626, 2003.

[114] N. Courty, E. Marchand, and B. Arnaldi, "Through-the-eyes control of a virtual humanoid," in *Computer Animation, 2001. The Fourteenth Conference on Computer Animation. Proceedings*, pp. 74–83, 2001.

[115] O. Lorch, A. Albert, J. Denk, M. Gerecke, R. Cupec, J. F. Seara, W. Gerth, and G. Schmidt, "Experiments in vision-guided biped walking," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3, pp. 2484–2490, IEEE, 2002.

[116] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade, "Vision-guided humanoid footstep planning for dynamic environments," *5th IEEE-RAS International Conference on Humanoid Robots, 2005.*, pp. 13–18.

[117] Y. Asano and A. Kawamura, "Stability on orientation motion of biped walking robot aiming at a target object," in *Advanced Motion Control, 2006. 9th IEEE International Workshop on*, pp. 428–432, IEEE, 2006.

[118] Y. Asano, "Decoupled rotational motion control for visual walking stabilization," *2008 10th IEEE International Workshop on Advanced Motion Control*, pp. 68–73, Mar. 2008.

[119] Y. Fujimoto, T. Imai, A. Kawamura, and Y. Asano, "Control of biped walking robot for human living environment," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 4, pp. 218–230, Mar. 2009.

[120] W. Song, T. Maeba, G. Wang, M. Minami, A. Yanou, and Y. Zhang, "Standing/walking stabilization of humanoid robot by visual servoing concept through online-visual-pose-estimation," in *SICE Annual Conference (SICE), 2011 Proceedings of*, pp. 911–916, IEEE, 2011.

[121] N. Oda and M. Ito, "Visual walking direction control by regulating torsional deflection for biped robot," in *Advanced Motion Control, 2010 11th IEEE International Workshop on*, pp. 143–148, IEEE, 2010.

[122] M. Marey and F. Chaumette, "A new large projection operator for the redundancy framework," *2010 IEEE International Conference on Robotics and Automation*, pp. 3727–3732, May 2010.

[123] K. Waldron and J. Schmiedeler, "Kinematics," *Springer Handbook of Robotics*, pp. 9–33, 2008.

[124] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, vol. 29. CRC Press, 1994.

[125] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, 1987.

[126] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*, vol. 3. Wiley New York, 2006.

[127] B. Bayle, M. Renaud, and J.-Y. Fourquet, "Nonholonomic mobile manipulators: kinematics, velocities and redundancies," *Journal of Intelligent and Robotic Systems*, vol. 36, no. 1, pp. 45–63, 2003.

[128] J. J. Craig, *Introduction to robotics: mechanics and control*. Pearson/Prentice Hall Upper Saddle River, NJ, USA:, 2005.

[129] Y. Hurmuzlu, F. Génot, and B. Brogliato, "Modeling, stability and control of biped robots - a general framework," *Automatica*, vol. 40, no. 10, pp. 1647–1664, 2004.

[130] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. Springer, 2000.

[131] Y. Hurmuzlu and C. Basdogan, "On the measurement of dynamic stability of human locomotion," *Journal of biomechanical engineering*, vol. 116, no. 1, pp. 30–36, 1994.

[132] F. Génot, *Contributions a la modélisation et a la commande des systèmes mécaniques de corps rigides avec contraintes unilatérales*. PhD thesis, Institut National Polytechnique de Grenoble-INPG, 1998.

[133] P.-B. Wieber, "On the stability of walking systems," in *Proceedings of the international workshop on humanoid and human friendly robotics*, 2002.

[134] Y. Hurmuzlu and D. B. Marghitu, "Rigid body collisions of planar kinematic chains with multiple contact points," *The International Journal of Robotics Research*, vol. 13, no. 1, pp. 82–92, 1994.

[135] S. Kajita and B. Espiau, "Legged robots," *Springer handbook of robotics*, pp. 361–389, 2008.

[136] S. Kajita and K. Tani, "Experimental study of biped dynamic walking in the linear inverted pendulum mode," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 3, pp. 2885–2891, IEEE, 1995.

[137] S. Kajita, O. Matsumoto, and M. Saigo, "Real-time 3D walking pattern generation for a biped robot with telescopic legs," *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 3, pp. 2299–2306, 2001.

[138] T. Sugihara, Y. Nakamura, and H. Inoue, "Real-time humanoid motion generation through zmp manipulation based on inverted pendulum control," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2, pp. 1404–1409, IEEE, 2002.

[139] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture Point: A Step toward Humanoid Push Recovery," *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pp. 200–207, Dec. 2006.

[140] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, and H. Hirukawa, "A real-time pattern generator for biped walking," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1, pp. 31–37, IEEE, 2002.

[141] T. Katayama, T. Ohki, T. Inoue, and T. Kato, "Design of an optimal controller for a discrete-time system subject to previewable demand," *International Journal of Control*, vol. 41, no. 3, pp. 677–699, 1985.

[142] P.-b. Wieber, "Trajectory Free Linear Model Predictive Control for Stable Walking in the Presence of Strong Perturbations," *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pp. 137–142, Dec. 2006.

[143] H. Diedam, D. Dimitrov, P.-B. Wieber, K. Mombaur, and M. Diehl, "Online walking gait generation with adaptive foot positioning through Linear Model Predictive control," *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1121–1126, Sept. 2008.

[144] Aldebaran-robotics, "NAO Technical overview, hardware: Links V4.0." https://community.aldebaran-robotics.com/doc/1-14/family/robots/links-robot.html, 2015. Retrieved March 25, 2015.

[145] M. Bombile, "Visual servoing based positioning and object tracking on humanoid robot," in *New Trends in Networking, Computing, E-learning, Systems Sciences, and Engineering*, pp. 19–27, Springer, 2015.

[146] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Advanced Robotics, 1991.'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pp. 1211–1216, IEEE, 1991.

[147] O. Khatib, "The impact of redundancy on the dynamic performance of robots," *Laboratory Robotics and Automation*, vol. 8, no. 1, pp. 37–48, 1996.

[148] A. Liegois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 7, pp. 868–871, Dec. 1977.

[149] T. Yoshikawa, "Manipulability of robotic mechanisms," *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[150] E. Marchand, F. Chaumette, and A. Rizzo, "Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing," in *Intelligent Robots and Systems' 96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, vol. 3, pp. 1083–1090, IEEE, 1996.

[151] F. Chaumette and T. Marchand, "A redundancy-based iterative approach for avoiding joint limits: application to visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 5, pp. 719–730, 2001.

[152] M. Marey and F. Chaumette, "New strategies for avoiding robot joint limits: Application to visual servoing using a large projection operator," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 6222–6227, 2010.

[153] N. Mansard and F. Chaumette, "Task sequencing for high-level sensor-based control," *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 60–72, 2007.

[154] N. Mansard, a. Remazeilles, and F. Chaumette, "Continuity of Varying-Feature-Set Control Laws," *IEEE Transactions on Automatic Control*, vol. 54, pp. 2493–2505, Nov. 2009.

[155] R. Penrose, "A generalized inverse for matrices," in *Mathematical proceedings of the Cambridge philosophical society*, vol. 51, pp. 406–413, Cambridge Univ Press, 1955.

[156] E. Marchand, A. Rizzo, and F. Chaumette, "Avoiding robot joint limits and kinematic singularities in visual servoing," in *Pattern Recognition, International Conference on*, vol. 1, pp. 297–297, IEEE Computer Society, 1996.

[157] F. Chaumette and E. Marchand, "A new redundancy-based iterative scheme for avoiding joint limits. application to visual servoing," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2, pp. 1720–1725, IEEE, 2000.

[158] F. Chaumette and É. Marchand, "A redundancy-based iterative approach for avoiding joint limits: Application to visual servoing," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 5, pp. 719–730, 2001.

[159] M. Vukobratovic, V. Potkonjak, and S. Tzafestas, "Human and Humanoid Dynamics," *Journal of Intelligent and Robotic Systems*, vol. 41, pp. 65–84, Sept. 2004.

[160] S. Kajita, F. Kanehiro, K. Kando, K. Yokoi, and H. Hirukawa, "The 3D Linear Inverted Pendulum Mode: A simple modeling for a biped walking pattern generation," pp. 239–246, 2001.

[161] M. Gienger, H. Janssen, and C. Goerick, "Task-oriented whole body motion for humanoid robots," in *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pp. 238–244, 2005.

[162] F. Chaumette, P. Rives, and B. Espiau, "Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 2248–2253, IEEE, 1991.

[163] F. Bensalah and F. Chaumette, "Compensation of abrupt motion changes in target tracking by visual servoing," in *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, vol. 1, pp. 181–187, IEEE, 1995.

[164] Y. Hu, R. Eagleson, and M. Goodale, "Human visual servoing for reaching and grasping: the role of 3D geometric features," *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 4, no. May, pp. 3209–3216, 1999.

[165] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "The nao humanoid: a combination of performance and affordability," *CoRR abs/0807.3223*, 2008.

[166] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "Mechatronic design of nao humanoid," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 769–774, IEEE, 2009.

[167] D. Gouaillier, C. Collette, and C. Kilner, "Omni-directional closed-loop walk for nao," in *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, pp. 448–454, IEEE, 2010.

[168] Aldebaran-robotics, "NAO Technical overview, Software." http://doc.aldebaran.com/1-14/getting-started/software-in-and-out.html, 2015. Retrieved March 25, 2015.

[169] R. Featherstone and D. E. Orin, "Dynamics," *Springer Handbook of Robotics*, pp. 35–65, 2008.

[170] N. Kofinas, E. Orfanoudakis, and M. Lagoudakis, "Complete analytical inverse kinematics for nao," in *Autonomous Robot Systems (Robotica), 2013 13th International Conference on*, pp. 1–6, April 2013.

[171] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[172] É. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: a generic software platform with a wide class of robot control skills," *Robotics & Automation Magazine, IEEE*, vol. 12, no. 4, pp. 40–52, 2005.

[173] Marchand, Eric and Spindler, Fabien, "Kalman Filter with Constant Acceleration and Colored Noise Model." http://www.irisa.fr/lagadic/visp/documentation/visp-2.10.0/classvpLinearKalmanFilterInstantiation.html, 2015. Retrieved April 06, 2015.

[174] A. J. Koivo and N. Houshangi, "Real-time vision feedback for servoing robotic manipulator with self-tuning controller," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 21, no. 1, pp. 134–142, 1991.

[175] N. Papanikolopoulos, P. Khosla, and T. Kanada, "Vision and control techniques for robotic visual tracking," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 857–864, IEEE, 1991.

[176] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision," *Robotics and Automation, IEEE Transactions on*, vol. 9, no. 1, pp. 14–35, 1993.

[177] K. Hashimoto, T. Kimoto, T. Ebine, and H. Kimura, "Manipulator control with image-based visual servo," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 2267–2271 vol.3, Apr. 1991.

[178] K. Hashimoto, T. Ebine, and H. Kimura, "Dynamic visual feedback control for a hand-eye manipulator," in *Intelligent Robots and Systems, 1992., Proceedings of the 1992 lEEE/RSJ International Conference on*, vol. 3, pp. 1863–1868, July 1992.

[179] K. Hashimoto, T. Ebine, and H. Kimura, "Visual servoing with hand-eye manipulator-optimal control approach," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 5, pp. 766–774, 1996.

[180] R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M. M. A. Sanchez, and J. Marescaux, "Active filtering of physiological motion in robotized surgery using predictive control," *Robotics, IEEE Transactions on*, vol. 21, no. 1, pp. 67–79, 2005.

[181] J. Gangloff, R. Ginhoux, M. D. Mathelin, L. Soler, and J. Marescaux, "Model predictive control for compensation of cyclic organ motions in teleoperated laparoscopic surgery," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 2, pp. 235–246, 2006.

[182] J. A. Gangloff and M. F. de Mathelin, "High-speed visual servoing of a 6-dof manipulator using multivariable predictive control," *Advanced Robotics*, vol. 17, no. 10, pp. 993–1021, 2003.

[183] K. Hashimoto and H. Kimura, "Visual servoing with nonlinear observer," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 1, pp. 484–489, IEEE, 1995.

[184] R. Kelly, "Robust asymptotically stable visual servoing of planar robots," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 5, pp. 759–766, 1996.

[185] R. Kelly, R. Carelli, O. Nasisi, B. Kuchen, and F. Reyes, "Stable visual servoing of camera-in-hand robotic systems," *Mechatronics, IEEE/ASME Transactions on*, vol. 5, no. 1, pp. 39–48, 2000.

[186] R. Kelly, V. S. Davila, and J. A. L. Perez, *Control of robot manipulators in joint space*. Springer, 2006.

[187] S.-H. Hsu, "Motion control," in *Springer Handbook of Robotics*, pp. 133–159, Springer, 2008.

[188] J. Pomares, I. Perea, and F. Torres, "Dynamic visual servoing with chaos control for redundant robots," *Mechatronics, IEEE/ASME Transactions on*, vol. 19, no. 2, pp. 423–431, 2014.

[189] E. Zergeroglu, D. Dawson, M. de Queiroz, and S. Nagarkatti, "Robust visual-servo control of robot manipulators in the presence of uncertainty," in *Decision and Control, 1999. Proceedings of the 38th IEEE Conference on*, vol. 4, pp. 4137–4142, IEEE, 1999.

[190] E. Zergeroglu, D. M. Dawson, M. de Querioz, and A. Behal, "Vision-based nonlinear tracking controllers with uncertain robot-camera parameters," *Mechatronics, IEEE/ASME Transactions on*, vol. 6, no. 3, pp. 322–337, 2001.

[191] E. Zergeroglu, D. M. Dawson, M. S. de Queiroz, and P. Setlur, "Robust visual-servo control of robot manipulators in the presence of uncertainty," *Journal of Robotic Systems*, vol. 20, no. 2, pp. 93–106, 2003.

[192] V. Andaluz, R. Carelli, L. Salinas, J. M. Toibero, and F. Roberti, "Visual control with adaptive dynamical compensation for 3D target tracking by mobile manipulators," *Mechatronics*, vol. 22, pp. 491–502, June 2012.

[193] C. C. Cheah, "Adaptive Tracking Control for Robots with Unknown Kinematic and Dynamic Properties," *The International Journal of Robotics Research*, vol. 25, pp. 283–296, Mar. 2006.

[194] C.-C. Cheah, C. Liu, and J. Slotine, "Adaptive vision based tracking control of robots with uncertainty in depth information," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 2817–2822, IEEE, 2007.

[195] C.-S. Kim, E.-J. Mo, S.-M. Han, M.-S. Jie, and K.-W. Lee, "Image-based robust control of robot manipulators with image Jacobian and dynamics uncertainties," *2008 IEEE International Conference on Automation Science and Engineering*, pp. 732–737, Aug. 2008.

[196] C.-S. Kim, E.-J. Mo, S.-M. Han, M.-S. Jie, and K.-W. Lee, "Robust visual servo control of robot manipulators with uncertain dynamics and camera parameters," *International Journal of Control, Automation and Systems*, vol. 8, pp. 308–313, Apr. 2010.

[197] F. Li and H.-L. Xie, "Sliding mode variable structure control for visual servoing system," *International Journal of Automation and Computing*, vol. 7, pp. 317–323, Aug. 2010.

[198] M. Keshmiri, W.-F. Xie, and A. Mohebbi, "Augmented Image-Based Visual Servoing of a Manipulator Using Acceleration Command," *IEEE Transactions on Industrial Electronics*, vol. 61, pp. 5444–5452, Oct. 2014.

[199] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational Space Control: A Theoretical and Empirical Comparison," *The International Journal of Robotics Research*, vol. 27, pp. 737–757, June 2008.

[200] V. Andaluz, F. Roberti, and R. Carelli, "Robust control with redundancy resolution and dynamic compensation for mobile manipulators," in *Industrial Technology (ICIT), 2010 IEEE International Conference on*, pp. 1469–1474, IEEE, 2010.

[201] J.-J. E. Slotine, W. Li, *et al.*, *Applied nonlinear control*, vol. 199. Prentice-Hall Englewood Cliffs, NJ, 1991.

[202] H. K. Khalil and J. Grizzle, *Nonlinear systems*, vol. 3. Prentice hall Upper Saddle River, 2002.

[203] F. Bensalah and F. Chaumette, "Real time visual tracking using the generalized likelihood ratio test," in *Int. Conf. on Automation, Robotics and Computer Vision, ICARCV'94*, vol. 2, (Singapore), pp. 1379–1383, Nov. 1994.

[204] Aldebaran-robotics, "NAO Technical overview, hardware: Masses V4.0." http://doc.aldebaran.com/1-14/family/robots/masses-robot.html, 2015. Retrieved March 25, 2015.

[205] Aldebaran-robotics, "NAO Technical overview, hardware: Motors V4.0." http://doc.aldebaran.com/1-14/family/robots/motors-robot.html, 2015. Retrieved March 25, 2015.

Appendix A

Walking constraints

A.1 Constraints on CoP

The constraints on the CoP within the convex hull can be written as

$$\begin{bmatrix} d_x(\theta) & d_y(\theta) \end{bmatrix} \begin{bmatrix} P^x - F^{x\_ref} \\ P^y - F^{y\_ref} \end{bmatrix} \leq b \tag{A.1}$$

where $d_x(\theta)$ and $d_y(\theta)$ are respectively the x and y components of the normal to the edge and b represents the bound of the convex hull in the direction of the normal.

Writing Equation (A.1) over the prediction horizon gives

$$D_{k+1} \begin{bmatrix} P^x_{k+1} - F^{x\_ref}_{k+1} \\ P^y_{k+1} - F^{y\_ref}_{k+1} \end{bmatrix} \leq \boldsymbol{b}_{k+1} \tag{A.2}$$

with

$$\begin{aligned} \boldsymbol{F}^{h\_ref}_{k+1} &= \left(V^c_{k+1} + V^f_{k+1}\mathbf{1}_3\right) f^h_k + V^f_{k+1}.\mathbf{R}_x.\triangle\mathbf{F}_{s_i} \\ P^h_{k+1} &= S_{p_h}\hat{c}^h_k + U_{p_h}\dddot{C}^h_k \end{aligned} \tag{A.3}$$

Substituting (A.3) in (A.2) gives

$$\begin{aligned} D_{k+1} \begin{bmatrix} U_{p_h} & 0 & -V^f_{k+1}.\mathbf{R}_x \\ 0 & U_{p_h} & -V^f_{k+1}.\mathbf{R}_y \end{bmatrix} \underbrace{\begin{bmatrix} \dddot{C}^x_k \\ \dddot{C}^y_k \\ \triangle\mathbf{F}_{s_i} \end{bmatrix}}_{u_k} \\ \leq b_{k+1} + D_{k+1} \begin{bmatrix} \left(V^c_{k+1} + V^f_{k+1}\mathbf{1}_3\right) f^x_k - S_{p_h}\hat{c}^x_k \\ \left(V^c_{k+1} + V^f_{k+1}\mathbf{1}_3\right) f^y_k - S_{p_h}\hat{c}^y_k \end{bmatrix} \end{aligned} \tag{A.4}$$

A.1.1 Considered Case: Two Walking Cycles ($m = 3$)

The initial values of the cyclic vector V^c_{k+1} and matrix V^f_{k+1} are

$$V^c_{k+1} = \left[\begin{array}{cccc|cccc|cccc|cccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]^T \tag{A.5}$$

$$V^f_{k+1} = \left[\begin{array}{cccc|cccc|cccc|cccc} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right]^T \tag{A.6}$$

Considering a foot of i vertices, for instance $i = 4$ (rectangular foot), the normal vectors to those vertices over two cycles can be computed as

$$\overrightarrow{\boldsymbol{n}_\theta} = \left[\begin{array}{cc} cos\theta_{s_0} & -sin\theta_{s_0} \\ sin\theta_{s_0} & cos\theta_{s_0} \\ \hline cos\theta_{s_1} & -sin\theta_{s_1} \\ sin\theta_{s_1} & cos\theta_{s_1} \\ \hline cos\theta_{s_2} & -sin\theta_{s_2} \\ sin\theta_{s_2} & cos\theta_{s_2} \\ \hline cos\theta_{s_3} & -sin\theta_{s_3} \\ sin\theta_{s_3} & cos\theta_{s_3} \end{array}\right] \left[\begin{array}{cccc} \overrightarrow{n_1} & \overrightarrow{n_2} & \cdots & \overrightarrow{n_{n_e}} \end{array}\right] \text{ with } \overrightarrow{n_i} = \left[\begin{array}{c} n_{ix} \\ n_{iy} \end{array}\right] \tag{A.7}$$

The x or y components of $\overrightarrow{\boldsymbol{n}_\theta}$ can be written as ($h = x,\ y$)

$$\boldsymbol{n}_{\theta_i h} = \left[\begin{array}{c} n_{\theta 1 h} \\ n_{\theta 2 h} \\ \vdots \\ n_{\theta n_e h} \end{array}\right] \tag{A.8}$$

Over two cycles corresponding to the predicted horizon N_p, the normal vectors can be written as follows

$$D_{k+1} = \left[\begin{array}{cccccccc} M_x(\theta_0) & 0 & 0 & 0 & M_y(\theta_0) & 0 & 0 & 0 \\ 0 & M_x(\theta_1) & 0 & 0 & 0 & M_y(\theta_1) & 0 & 0 \\ 0 & 0 & M_x(\theta_2) & 0 & 0 & 0 & M_y(\theta_2) & 0 \\ 0 & 0 & 0 & M_x(\theta_3) & 0 & 0 & 0 & M_y(\theta_3) \end{array}\right] \tag{A.9}$$

with $M_h(\theta_i)$ the normal vectors over one step period given by

$$M_h(\theta_i) = \left[\begin{array}{cccc} \boldsymbol{n}_{\theta_i h} & 0 & 0 & 0 \\ 0 & \boldsymbol{n}_{\theta_i h} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \boldsymbol{n}_{\theta_i h} \end{array}\right] \in \mathbb{R}^{n_e \cdot \frac{N_p}{4} \times \frac{N_p}{4}} \tag{A.10}$$

The constraints $\boldsymbol{b}_i$ representing the maximum allowable difference between the reference and the actual CoP in the direction of the vertex i are given by

$$\boldsymbol{b}_{k+1} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_{N_p} \end{bmatrix} \text{ with } \boldsymbol{b}_i = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n_e} \end{bmatrix} \tag{A.11}$$

A.2 Constraints on Footsteps placements

These are formulated as linear constraints, they prevent overstretching or collisions of feet.

A.2.1 Translations

For positive reference velocity ($V > 0$) we have

$$\underbrace{\left[\begin{array}{cccccc} 0 & 0 & \cdots & 0 & 0 \\ 0 & & & & 0 \\ 0 & & & & 0 \\ & & & & \\ \vdots & & \ddots & & \vdots \\ & & & & \\ 0 & & & & 0 \\ 0 & & & & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{array}\right]}_{2\cdot N_c} \underbrace{\left[\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{array}\right]}_{2\cdot m} \left[\begin{array}{c} \dddot{C}_k^x \\ \dddot{C}_k^y \\ \hline \triangle \mathbf{F}_{s_i} \end{array}\right] \leq b_{st} \tag{A.12}$$

For negative reference velocity ($V < 0$) we have

$$\underbrace{\left[\begin{array}{cccccc} 0 & 0 & \cdots & 0 & 0 \\ 0 & & & & 0 \\ 0 & & & & 0 \\ & & & & \\ \vdots & & \ddots & & \vdots \\ & & & & \\ 0 & & & & 0 \\ 0 & & & & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{array}\right]}_{2\cdot N_c} \underbrace{\left[\begin{array}{cccccc} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{array}\right]}_{2\cdot m} \left[\begin{array}{c} \dddot{C}_k^x \\ \dddot{C}_k^y \\ \hline \triangle \mathbf{F}_{s_i} \end{array}\right] \leq b_{st} \tag{A.13}$$

with $b_{st} = b_{right\_st}$ when the robot stands on the right foot or $b_{st} = b_{left\_st}$ when the stance foot is left.

$$b_{right\_st} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} l_x - \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} l_{yi} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} l_{yo} \qquad b_{left\_st} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} l_x - \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} l_{yi} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} l_{yo} \tag{A.14}$$

with l_x the forward maximum step's length when $V > 0$ or the backward maximum step's length when $V < 0$, l_{yi} is the lateral minimum inter-feet distance and l_{yo} is the lateral maximum inter-feet distance.

A.2.2 Rotation

The constraints on the stance foot orientation are written as

$$\underbrace{\left[\begin{array}{ccc|ccc} 0 & \cdots & 0 & 1 & 0 & 0 \\ & & & -1 & 1 & 0 \\ & & & 0 & -1 & 1 \\ \vdots & \ddots & \vdots & -1 & 0 & 0 \\ & & & 0 & -1 & 0 \\ 0 & \cdots & 0 & 0 & 0 & -1 \end{array}\right]}_{N_c} \begin{bmatrix} \dddot{C}_k^{\theta} \\ \theta_{k+1}^{f} \end{bmatrix} \leq \boldsymbol{b}_{\theta} \tag{A.15}$$

with

$$b_{\theta} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \triangle\theta_{max} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} {}^{w}\theta_k^c \tag{A.16}$$

where $\triangle\theta_{max}$ is the maximum allowable change of stance foot orientation between successive steps and ${}^{w}\theta_k^c$ represents the current stance foot orientation

A.3 Swing Foot trajectories

A.3.1 X and Y Trajectories

Using cubic polynomial of the form

$$x(t) = at^3 + bt^2 + ct + d \tag{A.17}$$

The x and y trajectories of the swing foot relative the the stance foot are computed as

$$\begin{cases} \Delta^{\alpha}(kT) & = a(iT)^3 + b(iT)^2 + c(iT) + d \\ \Delta^{\alpha}(0) & = 0 \\ \dot{\Delta^{\alpha}}(0) & = \Delta^{\alpha}_{k-\tau} \\ \Delta^{\alpha}(t_s) & = 0 \\ \dot{\Delta^{\alpha}}(t_s) & = \Delta^{\alpha}_{k+\tau} \end{cases} \tag{A.18}$$

$$\Delta^{\alpha}(iT) = -\frac{2(\Delta^{\alpha}_{k+\tau}(iT) - \Delta^{\alpha}_{k-\tau})}{t_s^3}(iT)^3 + \frac{3(\Delta^{\alpha}_{k+\tau}(iT) - \Delta^{\alpha}_{k-\tau})}{t_s^2}(iT)^2 + \Delta^{\alpha}_{k-\tau} \tag{A.19}$$

with $\alpha = x$, y, and

$$\begin{bmatrix} \Delta^{x}_{k-\tau} \\ \Delta^{y}_{k-\tau} \end{bmatrix} = \begin{bmatrix} \cos(\Delta^{\theta}_{k-\tau}) & \sin(\Delta^{\theta}_{k-\tau}) \\ -\sin(\Delta^{\theta}_{k-\tau}) & \cos(\Delta^{\theta}_{k-\tau}) \end{bmatrix} F^{s_0}_{s_1}((k-\tau)T) \tag{A.20}$$

where $\Delta^{\theta}_{k-\tau} = {}^{w}\theta^{c}_{k} - {}^{w}\theta^{c}_{k-\tau}$ is the difference between the current and the previous stance foot orientation and $F^{s_0}_{s_1}((k-\tau)T)$ is the first predicted step relative to the previous stance foot. $\tau = \frac{t_s}{T}$, is number of samples during a step period t_s such that $(k-\tau)$ is related to the previous step while $(k+\tau)$ is related to the future step. T is the period of the LMPC algorithm and k is the discrete time index of the control algorithm.

To determine $\Delta^{\alpha}_{k+\tau}(iT)$, let us consider the position of the swing foot relative to the stance foot which can be written as

$$T^{st\_ft}_{sw\_ft} = R^{st\_ft}_{CoM} T^{CoM}_{sw\_ft} + T^{st\_ft}_{CoM} \tag{A.21}$$

where the sub-scripts st_ft, sw_ft and CoM stand respectively for stance foot, swing foot and center of mass.

Thus at the end of a step period, the feet position relative to the CoM are

$$T^{st\_ft}_{CoM} = \begin{bmatrix} \frac{\Delta x^0_1}{2} \\ \frac{\Delta y^0_1}{2} \\ z_c \end{bmatrix}; \qquad T^{CoM}_{sw\_ft} = \begin{bmatrix} \frac{\Delta x^0_1}{2} \\ \frac{\Delta y^0_1}{2} \\ -z_c \end{bmatrix} \tag{A.22}$$

Substituting (A.22) in (A.21) gives

$$T^{st\_ft}_{sw\_ft}(t) = \begin{bmatrix} cos\Delta\theta(t) & -sin\Delta\theta(t) & 0 \\ sin\Delta\theta(t) & cos\Delta\theta(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\Delta x^0_1}{2} \\ \frac{\Delta y^0_1}{2} \\ -z_c \end{bmatrix} + \begin{bmatrix} \frac{\Delta x^0_1}{2} \\ \frac{\Delta y^0_1}{2} \\ z_c \end{bmatrix} \tag{A.23}$$

$$T_{sw\_ft}^{st\_ft}(t) = \begin{bmatrix} \left[\begin{pmatrix} cos\Delta\theta(t) & -sin\Delta\theta(t) \\ sin\Delta\theta(t) & cos\Delta\theta(t) \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \begin{bmatrix} \frac{\Delta x_1^0}{2} \\ \frac{\Delta y_1^0}{2} \end{bmatrix} \\ -z_c + z_c \end{bmatrix} \tag{A.24}$$

After simplifying and considering the variation of the CoM orientation relative to the swing foot ($-\theta_{step} \leq \theta(t) \leq 0$), we obtain in discrete form

$$\begin{bmatrix} \Delta_{k+\tau}^x(iT) \\ \Delta_{k+\tau}^y(iT) \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 + \cos(-\Delta_{k-\tau}^\theta + \Delta\theta(iT)) & \sin(-\Delta_{k-\tau}^\theta + \Delta\theta(iT)) \\ -\sin(-\Delta_{k-\tau}^\theta + \Delta\theta(iT)) & 1 + \cos(-\Delta_{k-\tau}^\theta + \Delta\theta(iT)) \end{bmatrix} F_{s_1}^{s_0}(k+\tau) \tag{A.25}$$

with

$$F_{s_1}^{s_0}(k+\tau) = 2 \begin{bmatrix} \frac{\Delta x_1^0}{2} \\ \frac{\Delta y_1^0}{2} \end{bmatrix} \tag{A.26}$$

and $\Delta\theta(iT)$ interpolated as

$$\Delta\theta(iT) = -\frac{2\Delta_{k-\tau}^\theta}{t_s^3}(iT)^3 + \frac{3\Delta_{k-\tau}^\theta}{t_s^2}(iT)^2 \tag{A.27}$$

A.3.2 Z trajectory

Using the fourth order polynomial of the form

$$z(t) = at^4 + bt^3 + ct^2 + dt + e \tag{A.28}$$

The z trajectory of the swing foot relative to the stance foot, while imposing a maximum height (z_{max}) and velocity constraints at the beginning and the end of a step is computed as

$$\begin{cases} \Delta^z(kT) & = a(iT)^3 + b(iT)^2 + c(iT) + d \\ \Delta^z(0) & = 0 \\ \dot{\Delta}^z(0) & = 0 \\ \Delta^z(\frac{t_s}{2}) & = z_{max} \\ \Delta^z(t_s) & = 0 \\ \dot{\Delta}^z(t_s) & = 0 \end{cases} \tag{A.29}$$

Which gives

$$\Delta^z(iT) = \frac{16z_{max}}{t_s^4}(iT)^4 - \frac{32z_{max}}{t_s^3}(iT)^3 + \frac{16z_{max}}{t_s^2}(iT)^2 \tag{A.30}$$

Appendix B

Kinematics of NAO Right Limbs

B.1 Denavit-Hartenberg Convention

According to the D-H convention, the frames are located with only four rather than six parameters. The joints of an $n-$ links manipulator are numbered from 1 to n, while the links are numbered from 0 to n starting from the base. As depicted in Figure B.1, the joint i connects the link $i-1$ to link i, and its location is considered fixed with respect to link $i-1$. The axis of revolute joint i is aligned with z_{i-1} axis. The x_i axis is directed along the common normal from z_{i-1} to z_i. In case of intersecting axes, x_i is usually taken parallel to $z_{i-1} \times z_i$. [31, pp. 7 - 10].

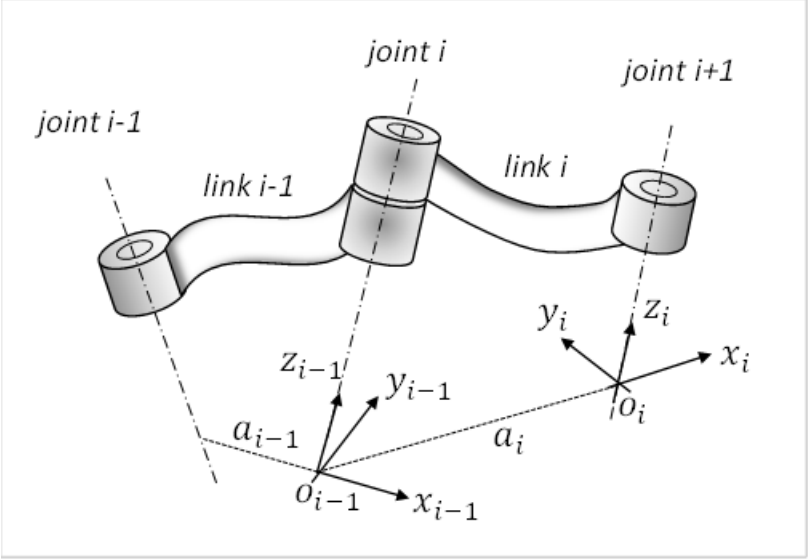


Figure B.1: Denavit-Hartenberg convention on the assignment of frames

As a result of this convention, the pose of the i^{th} link's frame with respect to the $i^{th}-1$ link's frame is given by a 4×4 homogeneous transformation matrix called D-H matrix

$$A_i^{i-1} = \begin{bmatrix} cos\theta_i & -cos\theta_i cos\alpha_i & cos\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i cos\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\theta_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{B.1}$$

This matrix derives from the product of four basic transformations

$$A_i^{i-1} = Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i} \tag{B.2}$$

where θ_i is the joint angle, d_i is the joint offset, a_i is the link length and α_i is the link twist (see ref. [126, p. 64] for more details).

B.2 Forward Kinematics

B.2.0.1 Forward kinematics of NAO's Right Arm

From the frames assignment in Figure 5.2, we derive the D-H parameters of NAO's right hand and summarized them in table (B.1).

Table B.1: DH. parameters of NAO's Right Arm

Link (joint)	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$	***Range (degrees)***
Base	T_{ZB}(ShouldderOffsetZ).T_{YB}(ShoulderOffsetY).R_{XB}(-90°)				
RshoulderPitch	0	90°	0	θ_1	-119.5-119.5
RshoulderRoll	$-a_2$	90°	0	$\theta_2 + 90$	-76 - 18
RElbowYaw	0	-90°	d_3	θ_3	-119.5-119.5
RElbowRoll	0	90°	0	θ_4	-2 - 88.5
RWristYaw	0	-90°	d_5	θ_5	-104 - 104.5
End Effector	R_{Z2}(-90).T_{Z5}(-HandOffsetZ).T_{Y5}(-HandOffsetY)				

where a_2, d_3, and d_5 are respectively the elbow lateral offset, the upper arm length, and the hand horizontal offset (see Figure 5.2). Substituting table (B.1)'s parameters in the D-H matrix (B.1) gives the following matrices

$$A_{31}^0 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{32}^1 = \begin{bmatrix} -s_2 & 0 & c_2 & -a_2 s_2 \\ c_2 & 0 & s_2 & -a_2 c_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{33}^2 = \begin{bmatrix} c_3 & 0 & -s_3 & 0 \\ s_3 & 0 & c_3 & 0 \\ 0 & -1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

$$A_{34}^3 = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{35}^4 = \begin{bmatrix} c_5 & 0 & -s_5 & 0 \\ s_5 & 0 & c_5 & 0 \\ 0 & -1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{B.3}$$

The transformation matrix from the frame $O_0x_0y_0z_0$ to the base frame $O_Bx_By_Bz_B$ and the transformation matrix from the end-effector's frame $O_Ex_Ey_Ez_E$ to the frame $O_5x_5y_5z_5$ are respectively given by

$$A_{30}^{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -t_y \\ 0 & -1 & 0 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{3E}^{5} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -h_x \\ 0 & 0 & 1 & -d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{B.4}$$

Finally, the homogeneous transformation T_{3E}^{B} giving the orientation and position of the right hand's end-effector frame with respect to the base coordinate frame is obtained from the product of homogenous transformations (B.3) and (B.4) as follows

$$T_{3E}^{B} = A_{30}^{B} A_{31}^{0} A_{32}^{1} A_{33}^{2} A_{34}^{3} A_{35}^{4} A_{3E}^{5} \tag{B.5}$$

The result is the same as in (5.9) - (5.12), except that a_2 has to be replaced by $-a_2$ and t_y by $-t_y$.

B.2.0.2 Forward kinematics of NAO's Right Leg

Similarly to the left leg, from the frames assigned to the right leg's chain (see Figure 5.2), the D-H parameters of NAO's rigt leg are derived and summarized in table (B.2).

Table B.2: DH. parameters of NAO's Right Leg

Link (joint)	$\boldsymbol{a}_i$	$\boldsymbol{\alpha}_i$	$\boldsymbol{d}_i$	$\boldsymbol{\theta}_i$	***Range (degrees)***
Base	T_{ZB}(HipOffsetZ).T_{YB}(HipOffsetY).R_{XB}(-135°)				
RHipYawPitch	0	-90°	0	$\theta_p - 90^o$	-65.62 - 42.44
RHipRoll	0	90°	0	$\theta_1 - 45^o$	-42.30 - 23.76
RHipPitch	$-T_{hl}$	0	0	θ_2	-101.54 - 27.82
RKneePitch	$-T_{bl}$	0	0	θ_3	-5.90 - 121.47
RAnklePitch	0	-90°	0	θ_4	-67.97 - 53.40
RAnkleRoll	$-F_{tl}$	0	0	θ_5	-45.03 - 22.27
End Effector	$R_{X5}(90^o).R_{Y5}(90^o)$				

where T_{hl}, T_{bl}, and F_{tl} are respectively the thigh length, the tibia length, and the foot height. Substituting table (B.2)'s parameters in the D-H matrix (B.1) gives the following matrices

$$A_{51}^{P} = \begin{bmatrix} c_{1+45^o} & 0 & s_{1+45^o} & 0 \\ s_{1+45^o} & 0 & -c_{1+45^o} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{52}^{1} = \begin{bmatrix} c_2 & -s_2 & 0 & -T_{hl}c_2 \\ s_2 & c_2 & 0 & -T_{hl}s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{53}^{2} = \begin{bmatrix} c_3 & -s_3 & 0 & -T_{bl}c_3 \\ s_3 & c_3 & 0 & -T_{bl}s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$A_{54}^{3} = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{55}^{4} = \begin{bmatrix} c_5 & -s_5 & 0 & -T_{bl}c_5 \\ s_5 & c_5 & 0 & -T_{bl}s_5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{B.6}$$

where c_{i+45^o}, and s_{i+45^o} refer to $cos(\theta_i + 45^o)$ and $sin(\theta_i + 45^o)$ respectively, while the superscript $_P$ stands for pelvis. The transformation matrix from the frame $O_0x_0y_0z_0$ to the base frame $O_Bx_By_Bz_B$, the transformation matrix from the pelvis frame $O_Px_Py_Pz_P$ to the frame $O_0x_0y_0z_0$, and the transformation matrix from the end-effector's frame $O_Ex_Ey_Ez_E$ to the frame $O_5x_5y_5z_5$ are respectively given by

$$A_{50}^{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{-45^o} & -s_{-45^o} & -H_{py} \\ 0 & s_{-45^o} & c_{-45^o} & -H_{pz} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{5P}^{0} = \begin{bmatrix} s_p & 0 & c_p & 0 \\ -c_p & 0 & s_p & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; A_{5E}^{5} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{B.7}$$

Finally, the homogeneous transformation $\mathbf{T}_{5E}^{B}$ giving the orientation and position of the left leg's end-effector frame with respect to the base coordinate frame is obtained from the product of homogenous transformations (B.6) and (B.7) as follows

$$\mathbf{T}_{5E}^{B} = A_{50}^{B} A_{5P}^{0} A_{51}^{P} A_{52}^{1} A_{53}^{2} A_{54}^{3} A_{55}^{4} A_{5E}^{5} \tag{B.8}$$

Using the definition () and the notation s_{ijk} , respectively c_{ijk} for $sin(\theta_i + \theta_j + \theta_k)$, respectively $cos(\theta_i + \theta_j + \theta_k)$, we obtain

B.2.1 Inverse Kinematics of NAO

B.2.1.1 Inverse Kinematics of NAO's Right Arm

The solution to the inverse kinematics of the right arm is derived by following the same procedure as for the left arm. The resulting close form solution are given by

$$\theta_1 = atan\left(\frac{-n_z(d_5 + h_x) + a_z d_z + p_z - t_z}{-n_x(d_5 + h_x) + a_x d_z + p_x}\right) \pm 180^o \tag{B.9}$$

$$\theta_2 = asin\left(\frac{-n_y(d_5 + h_x) + a_y d_z + p_y + t_y}{\sqrt{a_2^2 + d_3^2}}\right) + atan\left(\frac{a_2}{d_3}\right) \tag{B.10}$$

$$\theta_3 = atan\left(\frac{-n_x sin\theta_1 + n_z cos\theta_1}{sin\theta_2(n_x cos\theta_1 + n_z sin\theta_1) - n_y cos\theta_2}\right) \tag{B.11}$$

$$\theta_4 = acos\left(cos\theta_2(n_x cos\theta_1 + n_z sin\theta_1) + n_y sin\theta_2\right) \tag{B.12}$$

$$\theta_5 = atan\left(-\frac{cos\theta_2(a_x cos\theta_1 + a_z sin\theta_1) + a_y sin\theta_2}{cos\theta_2(o_x cos\theta_1 + o_z sin\theta_1) + o_y sin\theta_2}\right) \tag{B.13}$$

B.2.1.2 Inverse Kinematics of NAO's Right Leg

The inverse kinematics of the right leg is similar to that of the left leg, because of the symmetry about the sagital plane. The only difference lies in the transformation from the frame of the pelvis joint, shared by both leg, to the base frame. Given the pose of the right leg end-effector, which can be written in form of an homogenous transformation $\mathbf{T}_{5E}^{B}$ and related to the leg's joint angle by (B.5) recalled here

$$T_{5E}^{B} = A_{50}^{B} A_{51}^{0} A_{51}^{P} A_{52}^{1} A_{53}^{2} A_{54}^{3} A_{55}^{4} A_{5E}^{5} \tag{B.14}$$

The solution to the follows all procedures used for the left leg. Thus, from

$$\mathbf{T}_{43}^{1} = (A_{51}^{P})^{-1}(A_{5P}^{0})^{-1}(A_{50}^{B})^{-1}\mathbf{T}_{5E}^{B}(A_{5E}^{5})^{-1}(A_{55}^{4})^{-1}(A_{54}^{3})^{-1} = A_{52}^{1}A_{53}^{2}, \tag{B.15}$$

we obtain θ_3 as

$$\theta_3 = acos\left(\frac{(p_x + a_x F_{tl})^2 + (p_x + a_y F_{tl} + H_{py})^2 + (p_z + a_z F_{tl} + H_{pz})^2}{2T_{bl}T_{hl}}\right) \tag{B.16}$$

From the equality of the last columns of both sides of the equation

$$\mathbf{T}_{51}^{5} = (A_{52}^{1}A_{53}^{2}A_{54}^{3}A_{55}^{4})^{-1} = (A_{55}^{4})^{-1}(A_{54}^{3})^{-1}(A_{53}^{2})^{-1}(A_{52}^{1})^{-1}, \tag{B.17}$$

we can write

$$\begin{cases} {}_{1}^{5}P_x &= \frac{p_x(n_z o_y - n_y o_z) + (p_y + H_{py})(n_x o_z - n_z o_x) + (p_z + H_{pz})(n_y o_x - n_x o_y)}{n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y} \\ {}_{1}^{5}P_y &= \frac{p_x(n_z a_y - n_y a_z) + (p_y + H_{py})(n_x a_z - n_z a_x) + (p_z + H_{pz})(n_y a_x - n_x a_y)}{n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y} \\ {}_{1}^{5}P_z &= \frac{p_x(o_z a_y - o_y a_z) + (p_y + H_{py})(o_x a_z - o_z a_x) + (p_z + H_{pz})(o_y a_x - o_x a_y)}{n_x o_y a_z + n_y o_z a_x + n_z a_y o_x - n_z o_y a_x - o_z a_y n_x - a_z o_x n_y} \end{cases} \tag{B.18}$$

Similarly to the left leg case, we obtain θ_5 as follows

$$\theta_5 = atan\left(\frac{-[p_x(n_z a_y - n_y a_z) + (p_y + H_{py})(n_x a_z - n_z a_x) + (p_z + H_{pz})(n_y a_x - n_x a_y)]}{p_x(n_z o_y - n_y o_z) + (p_y + H_{py})(n_x o_z - n_z o_x) + (p_z + H_{pz})(n_y o_x - n_x o_y) - F_{tl} D_r}\right), \tag{B.19}$$

and θ_4 as

$$\theta_4 = atan\left(\frac{-(T_{hl}cos\theta_3 + T_{bl})T_{hl}sin\theta_3 \pm {}_{1}^{5}P_z\sqrt{T_{hl}^2 + T_{bl}^2 + 2cos\theta_3 T_{hl}T_{bl} - ({}_{1}^{5}P_z)^2}}{(T_{hl}cos\theta_3 + T_{bl})^2 - ({}_{1}^{5}P_z)^2}\right) \tag{B.20}$$

where ${}_{1}^{5}P_z$ is given by (B.18). Finally, from the equation

$$\mathbf{T}_{52}^{0} = (A_{5P}^{0})^{-1}(A_{50}^{B})^{-1}\mathbf{T}_{5E}^{B}(A_{5E}^{5})^{-1}(A_{55}^{4})^{-1}(A_{54}^{3})^{-1}(A_{53}^{2})^{-1}, \tag{B.21}$$

following the procedure used for the left leg, it can be shown that θ_P, θ_1 and θ_2 will be given as follows

$$\theta_P = atan\left(\frac{a_x tan\theta_5 - o_x}{\frac{\sqrt{2}}{2}[tan\theta_5(a_z - a_y) + (o_z - o_y)]}\right), \tag{B.22}$$

$$\theta_1 = acos\left(\frac{\sqrt{2}}{2}[cos\theta_5(o_y + o_z) - sin\theta_5(a_y + a_z)]\right) - \frac{\pi}{4}, \tag{B.23}$$

and

$$\theta_2 = atan\left(\frac{tan(\theta_3 + \theta_4)[cos\theta_5(a_y + a_z) + sin\theta_5(o_y + o_z)] + (n_y + n_z)}{tan(\theta_3 + \theta_4)(n_y + n_z) - [cos\theta_5(a_z + a_y) + sin\theta_5(o_z + o_y)]}\right) \tag{B.24}$$

B.2.2 Velocity Kinematics

B.2.2.1 NAO Right Arm's Jacobian

NAO's right arm being symmetrical to its left arm, the Jacobian of NAO's right arm is similar to the left arm Jacobian and has the following form

$$J_{RArm} = \begin{bmatrix} z_0 \times (o_E - o_0) & z_1 \times (o_E - o_1) & z_2 \times (o_E - o_2) & z_3 \times (o_E - o_3) & z_4 \times (o_E - o_4) \\ z_0 & z_1 & z_2 & z_3 & z_4 \end{bmatrix}. \tag{B.25}$$

Exploiting the forward kinematics, from $\mathbf{T}_{3E}^{B}$, the position of the right hand's end-effector is

$$o_E = \begin{bmatrix} [((-c_1s_2c_3 + s_1s_3)s_4 + c_1c_2c_4)(h_x + d_5) + \{[(-c_1s_2c_3 + s_1s_3)c_4 - c_1c_2s_4]s_5 \\ -(c_1s_2s_3 + s_1c_3)c_5\}d_z + c_1(c_2d_3 + s_2a_2)] \\ [(c_2c_3s_4 + s_2c_4)(h_x + d_5) + ((c_2c_3c_4 - s_2s_4)s_5 + c_2s_3c_5)d_z + s_2d_3 - c_2a_2 - t_y] \\ -[((s_1s_2c_3 + c_1s_3)s_4 - s_1c_2c_4)(h_x + d_5) - \{((s_1s_2c_3 + c_1s_3)c_4 + s_1c_2s_4)s_5 \\ +(s_1s_2s_3 - c_1c_3)c_5)d_z + s_1(c_2d_3 + s_2a_2) + t_z] \end{bmatrix}. \tag{B.26}$$

From homogenous transformations $\mathbf{T}_{30}^{B}$, $\mathbf{T}_{31}^{B}$, $\mathbf{T}_{32}^{B}$, $\mathbf{T}_{33}^{B}$ and $\mathbf{T}_{34}^{B}$, obtained by using (B.3) - (B.4), the positions of the frames' origins are obtained as

$$\boldsymbol{o}_0 = \boldsymbol{o}_1 = \begin{bmatrix} 0 \\ -t_y \\ t_z \end{bmatrix},\ \boldsymbol{o}_2 = \begin{bmatrix} a_2c_1s_2 \\ -a_2c_2 - t_y \\ a_2s_1s_2 + t_z \end{bmatrix},\ \text{and } \boldsymbol{o}_3 = \boldsymbol{o}_4 = \begin{bmatrix} d_3c_1c_2 + a_2c_1s_2 \\ d_3s_2 - a_2c_2 - t_y \\ d_3s_1c_2 + a_2s_1s_2 + t_z \end{bmatrix}, \tag{B.27}$$

while using the extracted rotation matrices $\mathbf{R}_{30}^{B}$, $\mathbf{R}_{31}^{B}$, $\mathbf{R}_{32}^{B}$, $\mathbf{R}_{33}^{B}$ and $\mathbf{R}_{34}^{B}$ and equation (5.60), the joints' axes of rotation are obtained as

$$z_0 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},\ z_1 = \begin{bmatrix} s_1 \\ 0 \\ -c_1 \end{bmatrix},\ z_2 = \begin{bmatrix} c_1c_2 \\ s_2 \\ s_1c_2 \end{bmatrix},\ z_3 = \begin{bmatrix} c_1s_2s_3 + s_1c_3 \\ -c_2s_3 \\ s_1s_2s_3 - c_1c_3 \end{bmatrix},$$

$$z_4 = \begin{bmatrix} (-c_1s_2s_3 + s_1c_3)s_4 + c_1c_2c_4 \\ c_2c_3c_4 + s_2c_4 \\ (-s_1s_2s_3 - c_1c_3)s_4 + s_1c_2c_4 \end{bmatrix} \tag{B.28}$$

Substituting (B.26)-(B.28) in (B.25) gives the column vectors of J_{RArm} as follows

$$J_{RA11} = \begin{bmatrix} [-(s_1s_2s_3 + c_1c_3)[(h_x + d_5)s_4 + d_z s_5 c_4] \\ +s_1[d_3c_2 + a_2s_2 - d_z(s_5c_2s_4 + c_5s_2s_3) + (h_x + d_5)c_2c_4] + d_z c_5 c_1 c_3] \\ 0 \\ [(c_1s_2c_3 - s_1s_3)[(h_x + d_5)s_4 + d_z s_5 c_4] \\ +c_1[-d_3c_2 - a_2s_2 + d_z(s_5c_2s_4 + c_5s_2s_3) - (h_x + d_5)c_2c_4] + d_z c_5 c_1 c_3] \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$J_{RA12} = \begin{bmatrix} c_1\{(h_x + d_5)(c_2c_3s_4 + s_2c_4) + d_z[s_5(c_2c_3c_4 - s_2s_4) + c_2s_3c_5] + d_3c_2 - a_2s_2\} \\ (h_x + d_5)(s_2c_3s_4 - c_2c_4) + d_z[s_5(s_2c_3c_4 + c_2s_4) + s_2s_3c_5] - d_3c_2 - a_2s_2 \\ s_1\{(h_x + d_5)(c_2c_3s_4 + s_2c_4) + d_z[s_5(c_2c_3c_4 - s_2s_4) + c_2s_3c_5] + d_3c_2 - a_2s_2\} \\ s_1 \\ 0 \\ -c_1 \end{bmatrix}$$

$$J_{RA13} = \begin{bmatrix} -(c_1s_2s_3 + s_1c_3)[(h_x + d_5)s_4 + d_z c_5 c_4] + d_z c_5(c_1s_2c_3 - s_1s_3)] \\ c_2[(h_x + d_5)s_3s_4 + d_z(s_5c_4s_3 - c_5c_3)] \\ -(s_1s_2s_3 - s_1c_3)[(h_x + d_5)s_4 + d_z s_5 c_4] + d_z c_5(s_1s_2c_3 + c_1s_3)] \\ c_1c_2 \\ s_2 \\ s_1c_2 \end{bmatrix}$$

$$J_{RA14} = \begin{bmatrix} (c_1s_2s_3 - s_1c_3)[(h_x + d_5)c_4 - d_z s_5 s_4] + c_1c_2[(h_x + d_5)s_4 + d_z s_5 c_4] \\ c_2c_3[-(h_x + d_5)c_4 + d_z s_5 s_4] + s_2[(h_x + d_5)s_4 + d_z s_5 c_4] \\ (s_1s_2c_3 + c_1s_3)[(h_x + d_5)c_4 - d_z s_5 s_4] + s_1c_2[(h_x + d_5)s_4 + d_z s_5 c_4] \\ c_1s_2s_3 + s_1c_3 \\ -c_2s_3 \\ s_1s_2s_3 - c_1c_3 \end{bmatrix}$$

$$J_{RA14} = \begin{bmatrix} -d_z[c_5c_4(s_1s_3 - c_1s_2c_3) + s_5(s_1c_3 + c_1s_2s_3) + c_5s_4c_1c_2] \\ d_z[c_5(s_2s_4 - c_2c_3c_4) + s_5c_2s_3] \\ d_z[c_5c_4(c_1s_3 + s_1s_2c_3) + s_5(c_1c_3 - s_1s_2s_3) + c_5s_4s_1c_2] \\ (-c_1s_2s_3 + s_1c_3)s_4 + c_1c_2c_4 \\ c_2c_3c_4 + s_2c_4 \\ (-s_1s_2s_3 - c_1c_3)s_4 + s_1c_2c_4 \end{bmatrix}.$$

Finally, $\mathbf{J}_{RArm}$ can be written as

$$\mathbf{J}_{RArm} = \begin{bmatrix} J_{RA11} & J_{RA12} & J_{RA13} & J_{RA14} & J_{RA15} \end{bmatrix} \tag{B.29}$$

B.2.2.2 NAO Right Leg's Jacobian

Similarly to the left leg chain, NAO's right leg's chain has also six joints with three of them intersecting at the hip and two at the ankle. Thus, its associated Jacobian will have the following form

$$\mathbf{J}_{RLeg} = \begin{bmatrix} z_0 \times (o_E - o_0) & z_P \times (o_E - o_P) & z_1 \times (o_E - o_1) & z_2 \times (o_E - o_2) & z_3 \times (o_E - o_3) & z_4 \times (o_E - o_4) \\ z_0 & z_p & z_1 & z_2 & z_3 & z_4 \end{bmatrix} \tag{B.30}$$

Using the forward kinematics, from $\mathbf{T}_{5E}^{B}$, the position of the leg's end-effector is

$$\boldsymbol{o}_E = \begin{bmatrix} -s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + s_{1+45^o}F_{tl}s_5] - c_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ \{-\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} - s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + s_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ -(c_ps_{1+45^o} + c_{1+45^o})F_{tl}s_5] - H_{py}\} \\ \{-\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} + s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ +(c_ps_{1+45^o} - c_{1+45^o})F_{tl}s_5] - H_{pz}\} \end{bmatrix}. \tag{B.31}$$

From the homogenous transformations $\mathbf{T}_{50}^{B}$, $\mathbf{T}_{5P}^{B}$, $\mathbf{T}_{51}^{B}$, $\mathbf{T}_{52}^{B}$, $\mathbf{T}_{53}^{B}$ and $\mathbf{T}_{54}^{B}$, derived from (B.6) - (B.7), the position vectors of the frames origins are obtained as follows

$$o_0 = o_P = o_1 = \begin{bmatrix} 0 \\ -H_{py} \\ -H_{pz} \end{bmatrix},\ o_2 = \begin{bmatrix} -(s_pc_{1+45^o}c_2 + c_ps_2)T_{hl} \\ \frac{\sqrt{2}}{2}[(c_pc_{1+45^o} - s_{1+45^o})c_2 - s_ps_2]T_{hl} - H_{py} \\ -\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} + s_{1+45^o})c_2 - s_ps_2]T_{hl} - H_{pz} \end{bmatrix}, \text{ and}$$

$$o_3 = o_4 = \begin{bmatrix} -s_pc_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2) - c_p(T_{bl}s_{23} + T_{hl}s_2) \\ \frac{\sqrt{2}}{2}[(c_pc_{1+45^o} - s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2) - s_p(T_{bl}s_{23} + T_{hl}s_2)] - H_{py} \\ -\frac{\sqrt{2}}{2}[(c_pc_{1+45^o} + s_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2) - s_p(T_{bl}s_{23} + T_{hl}s_2)] - H_{pz} \end{bmatrix}. \tag{B.32}$$

From (5.60) and the rotation matrices $\mathbf{R}_{50}^{B}$, $\mathbf{R}_{5P}^{B}$, $\mathbf{R}_{51}^{B}$, $\mathbf{R}_{52}^{B}$, $\mathbf{R}_{53}^{B}$ and $\mathbf{R}_{54}^{B}$, the joints' axes of rotation are obtained as

$$z_0 = \begin{bmatrix} 0 \\ \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix},\ z_1 = \begin{bmatrix} c_p \\ \frac{\sqrt{2}}{2}s_p \\ -\frac{\sqrt{2}}{2}s_p \end{bmatrix},\ z_1 = z_2 = z_3 = \begin{bmatrix} -s_ps_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_ps_{1+45^o} + c_{1+45^o}) \\ -\frac{\sqrt{2}}{2}(c_ps_{1+45^o} - c_{1+45^o}) \end{bmatrix},$$

$$z_4 = \begin{bmatrix} -s_ps_{1+45^o}s_{234} + c_pc_{234} \\ \frac{\sqrt{2}}{2}[s_{234}(c_pc_{1+45^o} - s_{1+45^o}) + s_pc_{234}] \\ -\frac{\sqrt{2}}{2}[s_{234}(c_pc_{1+45^o} + s_{1+45^o}) + s_pc_{234}] \end{bmatrix} \tag{B.33}$$

Substituting (B.26)-(B.28) in (B.30) gives the column vectors of $\mathbf{J}_{RLeg}$ as follows

$$J_{RL11} = \begin{bmatrix} -c_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + c_{1+45^o}F_{tl}s_5] + s_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) \\ -\frac{\sqrt{2}}{2}\{s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + c_{1+45^o}F_{tl}s_5] + c_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5)\} \\ \frac{\sqrt{2}}{2}\{s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) + c_{1+45^o}F_{tl}s_5] + c_p(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5)\} \\ 0 \\ \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix} \tag{B.34}$$

$$J_{RL12} = \begin{bmatrix} -s_p[c_{1+45^o}(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - s_{1+45^o}F_{tl}s_5] \\ \frac{\sqrt{2}}{2}[(c_ps_{1+45^o} + c_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - (c_ps_{1+45^o} - c_{1+45^o})s_5F_{tl}] \\ -\frac{\sqrt{2}}{2}[(c_ps_{1+45^o} - c_{1+45^o})(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) - (c_pc_{1+45^o} + s_{1+45^o})s_5F_{tl}] \\ c_p \\ \frac{\sqrt{2}}{2}s_p \\ -\frac{\sqrt{2}}{2}s_p \end{bmatrix} \tag{B.35}$$

$$J_{RL13} = \begin{bmatrix} s_p s_{1+45^o}(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) - c_p(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5) \\ -\frac{\sqrt{2}}{2}[(c_p s_{1+45^o} - c_{1+45^o})(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5)] \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})(T_{bl}s_{23} + T_{hl}s_2 + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + T_{hl}c_2 + F_{tl}c_{234}c_5)] \\ -s_p s_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) \\ -\frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) \end{bmatrix} \tag{B.36}$$

$$J_{RL14} = \begin{bmatrix} s_p s_{1+45^o}(T_{bl}s_{23} + F_{tl}s_{234}c_5) - c_p(T_{bl}c_{23} + F_{tl}c_{234}c_5) \\ -\frac{\sqrt{2}}{2}[(c_p s_{1+45^o} - c_{1+45^o})(T_{bl}s_{23} + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + F_{tl}c_{234}c_5)] \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})(T_{bl}s_{23} + F_{tl}s_{234}c_5) + s_p(T_{bl}c_{23} + F_{tl}c_{234}c_5)] \\ -s_p s_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) \\ -\frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) \end{bmatrix} \tag{B.37}$$

$$J_{RL15} = \begin{bmatrix} (s_p s_{1+45^o} s_{234} - c_p c_{234})F_{tl}c_5 \\ -\frac{\sqrt{2}}{2}[(c_p s_{1+45^o} - c_{1+45^o})s_{234} + s_p c_{234}]F_{tl}c_5 \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})s_{234} + s_p c_{234}]F_{tl}c_5 \\ -s_p s_{1+45^o} \\ \frac{\sqrt{2}}{2}(c_p s_{1+45^o} + c_{1+45^o}) \\ -\frac{\sqrt{2}}{2}(c_p s_{1+45^o} - c_{1+45^o}) \end{bmatrix} \tag{B.38}$$

$$J_{RL16} = \begin{bmatrix} [(s_p c_{1+45^o} c_{234} + c_p s_{234})s_5 - s_p c_{1+45^o} c_5]F_{tl} \\ \frac{\sqrt{2}}{2}\{[(c_p s_{1+45^o} - c_{1+45^o})c_{234} - s_p s_{234}]s_5 + (c_p s_{1+45^o} + c_{1+45^o})c_5]F_{tl} \\ \frac{\sqrt{2}}{2}[(c_p s_{1+45^o} + c_{1+45^o})c_{234} - s_p s_{234}]s_5 - (c_p c_{1+45^o} - s_{1+45^o})c_5]F_{tl} \\ -s_p s_{1+45^o} s_{234} + c_p c_{234} \\ \frac{\sqrt{2}}{2}[s_{234}(c_p c_{1+45^o} - s_{1+45^o}) + s_p c_{234}] \\ -\frac{\sqrt{2}}{2}[s_{234}(c_p c_{1+45^o} + s_{1+45^o}) + s_p c_{234}] \end{bmatrix} \tag{B.39}$$

$\mathbf{J}_{RLeg}$ is finally given by

$$\mathbf{J}_{RLeg} = \begin{bmatrix} J_{RL11} & J_{RL12} & J_{RL13} & J_{RL14} & J_{RL15} & J_{RL16} \end{bmatrix} \tag{B.40}$$

Appendix C

Visual Target Motion and Disturbance Estimation using Kalman Filter

C.1 Kalman Filter with Constant Acceleration and Colored Noise Model

This Kalman filter found in ViSP and used for the target motion estimation is formulated as follows [173]

The state equation model

$$\begin{cases} x(k+1) & = x(k) + \Delta t \cdot x(k) + \nu(k) \\ \nu(k+1) & = \rho\nu(k) & + w_1(k) \\ \dot{x}(k+1) & = \dot{x}(k) & + w_2(k) \end{cases} \tag{C.1}$$

Where the noise $w_1(k)$ and $w_2(k)$ are assumed to be zero-mean, white, mutually uncorrelated and stationary random variables with variance $\sigma^2_{Q_1}$ and $\sigma^2_{Q_2}$.

The state update equation is

$$x_k = F_{k-1}x_{k-1} + w_{k-1} \tag{C.2}$$

The transition matrix F is

$$F = \begin{bmatrix} 1 & 1 & \Delta t \\ 0 & \rho & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{C.3}$$

The state covariance matrix

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \sigma_{Q_1}^2 & 0 \\ 0 & 0 & \sigma_{Q_2}^2 \end{bmatrix} \tag{C.4}$$

The measurement model

$$z(k) = Hx(k) + r(k) \tag{C.5}$$

with $H = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$, $z(k)$ the measured velocity and $r(k)$ the measured noise, zero-mean white mutually uncorrelated uncorrelated and stationary random variable with variance σ_R^2. Its covariance matrix is

$$R = \begin{bmatrix} \sigma_R^2 \end{bmatrix} \tag{C.6}$$

The initial state vector is set as

$$x(0) = \begin{bmatrix} z(0) \\ 0 \\ 0 \end{bmatrix} \tag{C.7}$$

The initial prediction covariance matrix $P_{(0|0)}$ is given by

$$P_{(0|0)} = \begin{bmatrix} \sigma_R^2 & 0 & \frac{\sigma_R^2}{\Delta t} \\ 0 & \frac{\sigma_{Q_1}^2}{(1-\rho^2)} & \frac{-\rho\sigma_{Q_1}^2}{(1-\rho^2)\Delta t} \\ \frac{\sigma_R^2}{\Delta t} & \frac{-\rho\sigma_{Q_1}^2}{(1-\rho^2)\Delta t} & \frac{\left(2\sigma_R^2 + \frac{\sigma_{Q_1}^2}{(1-\rho^2)}\right)}{\Delta t} \end{bmatrix} \tag{C.8}$$

In the implementation the following values were used for tracking while walking:

- $\sigma_{Q_1} = \sigma_{Q_2} = 0.000005$
- $\sigma_R = 0.085$,
- $\rho = 0.6$,

For NAO's head tracking we used :

- $\sigma_{Q_1} = \sigma_{Q_2} = 0.002$
- $\sigma_R = 0.006$,
- $\rho = 0.7$

C.2 Visual Target Motion and Disturbance Compensation

When considering the object motion the task function derivative is given by

$$\dot{\boldsymbol{e}}(t) = \mathbf{L}_e \, {}^cV + \frac{\partial \boldsymbol{e}}{\partial t} \tag{C.9}$$

Separating known from unknown parameters of $\mathbf{L}_e$, ($\mathbf{L}_e = \widehat{\mathbf{L}}_e + \widetilde{\mathbf{L}}_e$) we can write

$$\dot{\boldsymbol{e}}(t) = \widehat{\mathbf{L}}_e \, {}^cV + D_e \tag{C.10}$$

with the uncertainty and disturbance lumped as

$$D_e = (\widetilde{\mathbf{L}}_e \, {}^cV + \frac{\partial \boldsymbol{e}}{\partial t}) \tag{C.11}$$

Thus, the control velocity giving an exponential decrease could be

$${}^cV = -\widehat{\mathbf{L}}_e^{\dagger}(\Lambda_p \boldsymbol{e}(t) + \widehat{D}_e) \tag{C.12}$$

C.2.1 Stability Analysis

Considering the candidate Lyapunov function $\mathcal{L} = \frac{1}{2}\boldsymbol{e}^T(t)\boldsymbol{e}(t)$, its time derivative is given by

$$\dot{\mathcal{L}} = \boldsymbol{e}^T(t)\dot{\boldsymbol{e}}(t) \tag{C.13}$$

Substituting the control velocity gives

$$\dot{\mathcal{L}} = -\boldsymbol{e}^T(t)\widehat{\mathbf{L}}_e\widehat{\mathbf{L}}_e^{\dagger}\left(\Lambda_p \boldsymbol{e}(t) + \widehat{D}_e\right) + \boldsymbol{e}^T(t)D_e \tag{C.14}$$

which can be rewritten as

$$\dot{\mathcal{L}} = -\Lambda_p \boldsymbol{e}^T(t)\widehat{\mathbf{L}}_e\widehat{\mathbf{L}}_e^{\dagger}\boldsymbol{e}(t) + \boldsymbol{e}^T(t)\left(D_e - \widehat{\mathbf{L}}_e\widehat{\mathbf{L}}_e^{\dagger}\widehat{D}_e\right) \tag{C.15}$$

Under the assumption that $\widehat{\mathbf{L}}_e\widehat{\mathbf{L}}_e^{\dagger} > 0$, $-\Lambda_p \boldsymbol{e}^T(t)\widehat{\mathbf{L}}_e\widehat{\mathbf{L}}_e^{\dagger}\boldsymbol{e}(t)$ is negative-definite, thus $\widehat{D}_e$ has to be designed such that $\dot{\mathcal{L}} < 0$.

C.2.2 Designing of the feed-forward control input

A choice of $\widehat{D}_e$ could be

$$\widehat{D}_e = \left(\widehat{\mathbf{L}}_e\widehat{\mathbf{L}}_e^{\dagger}\right)^{-1} D_e \tag{C.16}$$

However, this will require the exact knowledge of D_e which is the variable to be estimated, thus accounting for this uncertainty a robust option could be

$$\widehat{\overline{D}}_e = \begin{cases} k \left\| \widehat{D}_e \right\| \frac{\boldsymbol{e}(t)}{\|\boldsymbol{e}(t)\|} & ; if \ \|\boldsymbol{e}(t)\| > \varepsilon_o \\ k \left\| \widehat{D}_e \right\| \frac{\boldsymbol{e}(t)}{\varepsilon_o} & ; if \ \|\boldsymbol{e}(t)\| \leq \varepsilon_o \end{cases} \tag{C.17}$$

which yields at its maximum [126, p. 264]

$$\dot{\mathcal{L}} = -\Lambda_p \boldsymbol{e}^T(t) \widehat{\mathbf{L}}_e \widehat{\mathbf{L}}_e^{\dagger} \boldsymbol{e}(t) + \frac{k \left\| \widehat{D}_e \right\| \varepsilon_o}{2} \tag{C.18}$$

Therefore the boundary layer ε_o can be chosen arbitrarily small to reduce the effects of the disturbance.

In the implementation, $\widehat{D}_e$ is obtained as follows

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \dot{\boldsymbol{e}}(t) - \widehat{\mathbf{L}}_{\xi_{tsk}} {}^c\mathbf{W}_w \begin{bmatrix} {}^w\mathbf{J}_{c\_hd} & {}^w\mathbf{J}_{re\_lg} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_{nk} \\ \dot{\mathbf{q}}_{lg\_walk} \end{bmatrix} \tag{C.19}$$

(see Equations (4.48) and (4.43)) for the definitions of the symbols.

Finally, $\widehat{D}_e$ is obtained by Filtering $\widehat{\frac{\partial \boldsymbol{e}}{\partial t}}$.